IPA Better Life with IT

情報処理システム 高信頼化 教訓集

ITサービス編

独立行政法人 情報処理推進機構
社会基盤センター

別冊Ⅰ：障害対策手法

2020年3月16日 改訂

情報処理システム高信頼化教訓集（IT サービス編）
別冊Ⅰ：障害対策手法

独立行政法人情報処理推進機構

別冊Ⅰ

障害対策手法

別冊Ⅰ：障害対策手法　　目次

１．はじめに

この「障害対策手法」は、この教訓集の各教訓を実践するために必要な対策手法を整理したものである。その対策手法と教訓の関係一覧をガバナンス／マネジメント領域（表１－１）と、技術領域（表１－２）に分けて示す。対策手法の整理に際しては、過去に IPA で蓄積されたソフトウェア・エンジニアリング手法を優先して活用した。

この「障害対策手法」は、教訓に含まれる具体的な対策を実施する際に、対策の必要な背景等をより深く理解するために役立つ。また、教訓に含まれる対策以外の周辺対策・関連対策を調査、検討する際の参考としても活用することができる。

システム障害の再発防止に向けた対策は、組織のガバナンスやシステム開発プロジェクトのマネジメント、システムのアーキテクチャ、システム構築・運用の手法やプロセスの継続的な評価・改善等、多岐にわたる。このような多種多様な対策を総合的にまとめ、対策の範囲を広くカバーすることができれば、この総合された対策群が、システムの構築や運用段階で留意すべき事項、いわば「システム構築・運用ガイド」となり、システム障害を開発時から予防することに活用できる。

表１－１　対策手法と教訓の関係（ガバナンス／マネジメント領域）

◆ガバナンス／マネジメント領域

		障害対策手法	文献	教訓番号（○:主となる対策手法、△:関連する対策手法） G1	G2	G3	G4	G5	G6	G7	G8	G9	G10	G11	G12	G13	G14	G15	G16	G17	G18	G19	G20	G21
ガバナンス／マネジメント領域	1	ユーザ企業とベンダ企業の連携、合意形成																							
		1 ユーザ企業による要求品質の確保	文献1		○														○					
		2 機能要件の合意形成	文献2		○														○					
	2	ユーザ企業内の事業部門と情シス部門との連携		○		○																		
	3	共同センタ利用における ユーザ企業の連携、合意形成																						
		1 性能要件の合意形成						○																
		2 障害時発生時の合意形成									○													
	4	クラウドセンタと利用企業の連携、合意形成	文献3							○		○												
	5	障害管理の取組み	文献4				○		○	○	○	○	○	○						○				○
	6	障害再発防止のための組織的マネジメント	文献5												○	○	○						○	
	7	問題解決プロセス（共通フレーム2013）	文献6										○	○				○						
	8	プロセス改善	文献7				○	○	○			○										○		
	9	運用時の定量的信頼性向上方法	文献8															○	○		○			
技術領域	1	トレーサビリティ管理																						
		1 製品に関するトレーサビリティ ISO9001	文献9																					
		2 文書およびデータに関するトレーサビリティ	文献10																					
	2	「見える化」手法																						
		1 暗黙知の整備・有効活用	文献11																					
		2 俯瞰図	文献11																					
	3	要求獲得手法	文献12																					
	4	変更管理																						
		1 共通フレーム2013	文献6																					
		2 JIS Q20000-1サービスマネジメント	文献13																					
	5	フェールソフト																						
	6	テスト技法																						
		1 テスト環境のリスク管理																						
		2 シミュレーション手法																						
		3 テスト網羅性の高度化技法	文献14																					
	7	可用性管理																						
		1 システムの冗長化設計																						
		2 シングルポイントの洗出し																						
		3 障害運用マニュアルの整備と訓練																						
		4 可用性の注意点：フルメッシュ構成																						
	8	非機能要求グレード	文献15					△																
	9	仮想化技術	文献16																					
	10	レビュー手法	文献17						△															
	11	サイレント障害対策																						
	12	パッチ管理	文献18																					
	13	高回復力システム基盤導入	文献19					△																
	14	RDBシステム管理	文献20																					
	15	ヒューマンファクターズ					△		△															
	16	ディペンダビリティ																						
	17	原因不明時の対策																						
	18	周期処理の対策	文献21																					
	19	レースコンディションの対策	文献22																					

表１－２　対策手法と教訓の関係（技術領域）

◆技術領域

	障害対策手法	文献	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17	T18	T19	T20	T21	T22	T23	T24	T25	T26	T27	T28	T29	T30	T31	T32	T33
			教訓番号（○:主となる対策手法、△:関連する対策手法）																																
ガバナンス／マネジメント領域	1 ユーザ企業とベンダ企業の連携、合意形成																																		
	1 ユーザ企業による要求品質の確保	文献1												△																					
	2 機能要件の合意形成	文献2																																	
	2 ユーザ企業内の事業部門と情シス部門との連携																																		
	3 共同センタ利用におけるユーザ企業の連携、合意形成																																		
	1 性能要件の合意形成																																		
	2 障害時発生時の合意形成																																		
	4 クラウドセンタと利用企業の連携、合意形成	文献3																																	
	5 障害管理の取組み	文献4															△															△		
	6 障害再発防止のための組織的マネジメント	文献5																														△		
	7 問題解決プロセス（共通フレーム2013）	文献6																									△							
	8 プロセス改善	文献7																	△				△											
	9 運用時の定量的信頼性向上方法	文献8																							△									
技術領域	1 トレーサビリティ管理																																		
	1 製品に関するトレーサビリティ ISO9001	文献9												○								○							○					
	2 文書およびデータに関するトレーサビリティ	文献10					○									○				○														
	2 「見える化」手法																																		
	1 暗黙知の整備・有効活用	文献11			○																	○													
	2 俯瞰図	文献11		○																															
	3 要求獲得手法	文献12				○	○																			○									
	4 変更管理																																		
	1 共通フレーム2013	文献6				○	○			○						○	○											○		○	○		○	△	△
	2 JIS Q20000-1 サービスマネジメント	文献13				○	○			○						○	○											○		○	○		○	△	△
	5 フェールソフト		○																																
	6 テスト技法																																		
	1 テスト環境のリスク管理							○			○																								
	2 シミュレーション手法				○																														
	3 テスト網羅性の高度化技法	文献14									○				○															○					
	7 可用性管理																																		
	1 システムの冗長化設計		○	○					○		○	○						○													○			
	2 シングルポイントの洗出し								○					○																	○			
	3 障害運用マニュアルの整備と訓練								○					○					○													○		
	4 可用性の注意点：フルメッシュ構成											○																						
	8 非機能要求グレード	文献15							○	○			○				○			○				○	○									
	9 仮想化技術	文献16								○																									
	10 レビュー手法	文献17					○								○																				
	11 サイレント障害対策												○											○	○									
	12 パッチ管理	文献18						○										○																	
	13 高回復力システム基盤導入	文献19									○	○																							
	14 RDBシステム管理	文献20																			○														
	15 ヒューマンファクターズ																						○												
	16 ディペンダビリティ																									○									
	17 原因不明時の対策																										○								
	18 周期処理の対策	文献21																																○	
	19 レースコンディションの対策	文献22																																	○

対策手法と関連している文献一覧

ここで取り上げている文献は、対策手法と教訓の関係（ガバナンス／マネジメント領域）（表１－１）、対策手法と教訓の関係（技術領域）（表１－２）で紹介しているものである。一部文献が数カ所に記載されているのは、利便性を考慮し表との関係箇所を明確にするためである。また、章タイトルを表示しているものもある。

また、表１－１、表１－２と直接関係ない文献については、ページ毎にある脚注を参照願いたい。

（文献 1）IPA『経営者が参画する要求品質の確保
～超上流から攻める IT 化の勘どころ～　第 2 版』2006 年
https://www.ipa.go.jp/sec/publish/tn05-002.html

（文献 2）IPA『機能要件の合意形成ガイド（ver.1.0）
～「発注者ビューガイドライン ver.1.0」改訂版～』2010 年
https://www.ipa.go.jp/sec/softwareengineering/reports/20100331.html

（文献 3）経済産業省/IPA『情報システム調達のための技術参照モデル（TRM）平成 25 年度版　クラウドサービス編』2014 年
https://www.ipa.go.jp/files/000045525.pdf

（文献 4）IPA『「障害管理の取組みに関する調査」調査報告書』2012 年
https://www.ipa.go.jp/sec/softwareengineering/reports/20121105.html

（文献 5）IPA『情報システム障害の再発防止のための組織的マネジメントの調査 WG 報告書』　2012 年
https://www.ipa.go.jp/files/000004616.pdf

（文献 6）IPA『共通フレーム 2013
～経営者、業務部門とともに取組む「使える」システムの実現～』2013 年
https://www.ipa.go.jp/sec/publish/tn12-006.html

（文献 7）IPA『プロセス改善ナビゲーションガイド<虎の巻編>』2009 年
https://www.ipa.go.jp/sec/publish/tn08-009.html

（文献 8）IPA『「情報システム運用時の定量的信頼性向上方法」に関する調査報告書』2015 年
https://www.ipa.go.jp/files/000045091.pdf

（文献 9）一般財団法人日本規格協会
『対訳 ISO9001: 2008 品質マネジメントの国際規格　ポケット版』2009 年

（文献 10）IPA「第 5 章トレーサビリティ管理の手法」
『高信頼化ソフトウェアのための開発手法ガイドブック
～予防と検証の事例を中心に～』2011 年
https://www.ipa.go.jp/files/000005144.pdf

（文献 11）IPA『IT プロジェクトの「見える化」上流工程編』2007 年
https://www.ipa.go.jp/sec/publish/tn06-001.html

（文献 12）一般社団法人情報サービス産業協会 REBOK 企画 WG
『要求工学知識体系　第 1 版』2011 年

（文献 13）一般財団法人日本規格協会『JIS Q 20000-1　情報技術　サービスマネジメント』2012 年

（文献 14）IPA「第 6 章テスト網羅性の高度化技法」
『高信頼化ソフトウェアのための開発手法ガイドブック
～予防と検証の事例を中心に～』2011 年
https://www.ipa.go.jp/files/000005144.pdf

（文献 15）IPA『非機能要求グレード利用ガイド』2010 年
https://www.ipa.go.jp/sec/softwareengineering/reports/20100416.html

（文献 16）経済産業省/IPA「2.5.仮想化技術」
『情報システム調達のための技術参照モデル（TRM）平成 25 年度版
クラウドサービス編』2014 年
https://www.ipa.go.jp/files/000045525.pdf

（文献 17）IPA「3.2.2 レビュー手法の概要」
『高信頼化ソフトウェアのための開発手法ガイドブック
～予防と検証の事例を中心に～』2011 年
https://www.ipa.go.jp/files/000005144.pdf

（文献 18）米国国立標準技術研究所（NIST）
（翻訳監修）IPA/NRI セキュアテクノロジーズ（株）
『パッチおよび脆弱性管理プログラムの策定』2005 年
https://www.ipa.go.jp/files/000025330.pdf

（文献 19）IPA『高回復力システム基盤導入ガイド（概要編）』2012 年
https://www.ipa.go.jp/sec/softwareengineering/reports/20120508.html

（文献 20）IPA「7-2-基.RDB システム管理に関する知識」
『OSS モデルカリキュラムの学習ガイダンス』2008 年
https://www.ipa.go.jp/files/000018500.pdf

（文献 21）IPA『組込みソフトウェア向け設計ガイド　ESDR[事例編]』2012 年
https://www.ipa.go.jp/files/000005148.pdf

（文献 22）IPA「第 4 章　不測の事態対策　レースコンディション」『セキュアプログラミング講座』
https://www.ipa.go.jp/security/awareness/vendor/programmingv2/contents/c304.html

２．ガバナンス／マネジメント領域

ガバナンス／マネジメント領域の教訓における対策の手法と事例を説明する。

システム障害を減らすためには、組織のガバナンス／マネジメントでの取組みが重要であり、運用時の対策は勿論のこと、システム開発の段階から対策を講じることが必要である。特に事業や業務検討の始まりから要件定義までの「超上流」工程での対策が重要である。超上流工程とは、システム化の方向性、システム化計画、要件定義の各プロセスを経過し、要件定義書を作成するまでの工程である。この工程においては、要求を正確に定義・伝達することと、その責任分担を明確にすること等の要求品質管理がポイントとなる。

２．１　ユーザ企業とベンダ企業の連携、合意形成

システム障害を減らすためには、ユーザ企業とベンダ企業との連携が重要である。

開発案件の増加に伴って任せる業務が徐々に拡大し、要件定義や受入れテスト等、発注者としての役割を果たし切れなくなっていたユーザ企業は、得てして、システム障害に対しての対策が後手に回ることになる。システム障害を減らすため、ユーザ企業は、要件をベンダ企業に明確に提示し、その要件通りにシステムが構築されているかを確認することが必要である。

２．１．１　ユーザ企業による要求品質の確保

超上流工程での要求品質管理では、ユーザ企業（発注者）とベンダ企業（受注者）との双方が超上流工程をうまく進めるためのポイントを「原理原則 17 ヶ条」（文献 1）としてまとめている（表２．１．１－１）。[1]

表２．１．１－１　原理原則 17 ヶ条

原理原則[1]	ユーザとベンダの想いは相反する
原理原則[2]	取り決めは合意と承認によって成り立つ
原理原則[3]	プロジェクトの成否を左右する要件確定の先送りは厳禁である
原理原則[4]	ステークホルダ間の合意を得ないまま、次工程に入らない
原理原則[5]	多段階の見積りは双方のリスクを低減する
原理原則[6]	システム化実現の費用はソフトウェア開発だけではない
原理原則[7]	ライフサイクルコストを重視する
原理原則[8]	システム化の方針・狙いの周知徹底が成功の鍵となる
原理原則[9]	要件定義は発注者の責任である
原理原則[10]	要件定義書はバイブルであり、事あらばここへ立ち返るもの
原理原則[11]	優れた要件定義書とはシステム開発を精緻にあらわしたもの
原理原則[12]	表現されない要件はシステムとして実現されない
原理原則[13]	数値化されない要件は人によって基準が異なる
原理原則[14]	「今と同じ」という要件定義はありえない
原理原則[15]	要件定義は「使える」業務システムを定義すること
原理原則[16]	機能要求は膨張する。コスト、納期が抑制する
原理原則[17]	要件定義は説明責任を伴う合意成熟度のレベル

ユーザ企業（発注者）は、要件定義書の中身と受入れテストについての責任を持つ必要がある。要件定義があいまいであったり、検討不足のままベンダ企業（受注者）に開発を依頼したりした場合、その結果として、コスト増、納期遅れだけでなく、品質低下をも発生させる恐れがあることを自覚することが重要

[1] IPA『経営者が参画する要求品質の確保
～超上流から攻める IT 化の勘どころ～　第 2 版』2006 年　P.87

である。

また、要件定義書の作成はベンダ企業に手伝ってもらうこともあるが、ユーザ企業の責任とし、契約で明確にする。要件定義と設計以降で契約を分け、要件定義の契約はユーザ企業が最終的な責任を持つ。

■本手法に関連する対策事例

「教訓Ｇ２」「教訓Ｇ１６」「教訓Ｔ１２」の対策

２．１．２　機能要件の合意形成

ユーザ企業とベンダ企業との間で要件に関する誤解のないように合意形成を図る必要がある。そのためには次の作業が重要となる（文献 2)。

（1）言い切る／聞き切る

発注者は、次のような要件定義の結果を伝える。

- システム化の目的・範囲
- 運用時や保守拡張時の操作や業務
- 業務遂行において変えることのできない制約条件（業務や組織構造、連携システム等により生じる制約、及び業務上の法令や規則等）

これらを正しく伝えるために、発注者が言い切ること、開発者が聞き切ることが、「言い切る／聞き切る」ことになる。

（2）図表に書く

- 開発者は、設計書の記述上の約束事を共通ルールとして記述する。
- 開発者は、発注者から聞き切った情報を反映して書く。
- 開発者は、開発グループ内のレビューで指摘された欠陥を修正する。

（3）もれ／矛盾をチェックする

- 開発者は、共通ルール自体にもれ／矛盾がないことをチェックする。
- 開発者は、「工程成果物」が共通ルールに沿っていることをチェックする。
- 開発者は、他の技術領域の「工程成果物」を含め、整合をチェックする。

（4）一緒にレビューする（発注者と開発者で）

- 発注者の要件と開発者の設計に齟齬がないことをレビューする。
- 発注者は開発者が想定していない前提や運用がないかをレビューする。
- システム化の目的と範囲に照らし合わせて、「工程成果物」が正しく書かれていることをレビューする。

■本手法に関連する対策事例

「教訓Ｇ２」「教訓Ｇ１６」の対策

２．２　ユーザ企業内の事業部門と情シス部門との連携

システム障害を減らすためには、ユーザ企業内の事業部門と情シス部門との連携が重要である。

システム開発においては、情シス部門が主導権を持って、事業部門からの要件を引出し、要件定義書としてまとめるのが一般的である。しかし、事業部門がその要件を正確に情シス部門に伝えることができず、サービスイン後に、障害が多発する事例が存在する。

ここで、要件定義には、以下の 3 種類が存在する。

・事業要件定義

ビジネス環境を分析し、業務範囲や業務分掌等を定めた上で、必要な業務手順を決める。

・業務要件定義

業務内容、業務形態、業務品質、性能目標、運用、以降要件、セキュリティ等の業務モデルの検討を行う。

・システム要件定義

システム構成、DB・ファイル構造、運用、移行要件、セキュリティ、機密情報保護対策等のシステムモデルの検討を行う。

これら 3 種類の要件定義のうち、事業要件定義と業務要件定義は、明らかに事業部門の責任である。システム要件定義についても、事業部門が、情シス部門との対話を繰り返しながら、事業要件定義、業務要件定義と合致させていくため、事業部門が責任を有する。

また、事業部門は、要件定義どおりにシステムが出来たかどうか受入れテストを実施する責任を負う。

■本手法に関連する対策事例

「教訓Ｇ１」「教訓Ｇ３」の対策

２．３　共同センタ利用におけるユーザ企業の連携、合意形成

同業者同士が同一システムを共同利用し、それを運営するセンタを立ち上げる事例が、いくつかの産業分野で見受けられる。

例　・金融機関どうしの共同センタ
　　・自治体どうしの共同センタ
　　・鉄道会社が参画する IC 乗車券の共同管理センタ

ここでは、共同センタにおけるユーザ企業の連携が必要な、処理能力の問題、システム障害発生時の課題の２点についての対策を述べる。

２．３．１　性能要件の合意形成

共同センタにおける処理能力オーバ等のデータ量見積誤りにおける障害防止策について、運営する上での参加企業と運営センタの合意形成におけるポイントを以下に述べる。

（１）ピーク件数予測に対しては、参画企業が自社でコミットした予測を行うこと。１社でも、予測をいい加減に出すと、参画企業全体に迷惑を及ぼすことになる。

（２）上記ピーク件数予測に対してのリスク対策を、参画企業、運営会社ともに、共通の認識に立ち、行うこと。

（３）障害対策を運営会社だけに任せないこと。参画各社が、運営をベンダ任せにして、十分なコミットメントをもった「かまえ＝態勢」を構築すること。そのためには、参画各社で「推進協議会」を結成し、重要なシステム変更やリリースに関しては、この推進協議会でレビューを繰り返して判断する態勢とすること。

（４）各社がバックアップシステムの必要性を再認識し、別センタでバックアップシステムを構築すること。

このように共同システムであっても、また委託先がしっかりした運営をしているように見えても、委託する側の１社１社が自社のシステム運営と同様の責任をもって関わる（コミットする）態勢を構築することが重要である。

■本手法に関連する対策事例

「教訓Ｇ５」の対策

２．３．２　障害発生時の合意形成

システム基盤を共同利用している場合、ある会社のソフトウェアの障害復旧のために、同じシステム基盤の上で稼働している他社のサービスを止めざるを得ない事態が発生することがある。

このような事態を想定して、サービスの利用者同士、運営事業者が、障害発生時に備えた対策を事前に立てておくことが重要となる。

まず、障害発生時の関係者の役割分担や、利用者業務への影響範囲を明確化する。特に、障害発生時に停止／再起動する単位と、その停止により影響を受ける利用者の範囲を事前に整理・明確化する。

次に、システムを停止／再起動させる場合についての条件や手順、責任等に関する取決めを定義し、共同利用各社と合意する。

また、共同利用者間の日常の情報共有を行うと共に、非常時の緊急連絡体制等を定めておく。

その他、定期的に会合を持ち、順次、対策が必要な案件を率直に提示しながら、安定運用に向けた取組みを、サービスの利用者同士、運営事業者が一体となって行っていくことが必要である。

■本手法に関連する対策事例

「教訓Ｇ８」の対策

２．４　クラウドセンタと利用企業の連携、合意形成

今回、当対策集で述べるクラウドセンタは、インターネットを介して不特定多数のユーザを対象に提供されるパブリッククラウドではなく、企業が自組織内に閉じて利用するプライベートクラウドについての対策である。

IPA では、調達に必要な技術情報をまとめた「技術参照モデル（TRM：Technical Reference Model)」を提供し、政府の指針に従った公平な IT システムの調達を支援した。その中で、クラウドサービス編が、パブリッククラウドでのクラウドセンタと利用企業の連携、合意形成に役立つと思われるので、以下その概要を紹介する（文献 3)。

クラウドとは、一般的に「ネットワークを通じて、情報処理サービスを、必要に応じて提供／利用する」形の情報処理の仕組みであり、その存在が雲の中にあるようなイメージであることからクラウドと称されている。

クラウドセンタは、何が提供されるかは明示するものの、それがいかに実現されるかは隠蔽（カプセル化）しており、サービスを提供すること、サービス化がクラウドセンタの役割である。

サービス化とは、アプリケーションやプラットフォーム、インフラのような機能が、オーダーメイドではなく標準化、規格化されたレディーメイドな状態で提供され、使える機能が予めメニュー化、可視化されていることをさす。そして、その使い方が明示されており、セルフサービスで使えることや、オンデマンド化されて、すぐに使える状態であること、また、そのサービスレベルが SLA によって予め定義されていることも重要な要素である。

クラウドサービスでは、障害や事故が発生した際の影響が大きいため一層の高信頼設計が必要であること等、集約による考慮すべき点がある。一元管理、自動化、オンデマンド化等の実現レベルがコスト削減と迅速性の双方からクラウド化における長短があることを認識して、クラウドの利用や構築に臨まれたい。

以下、クラウドサービスを利用するにあたり、考慮すべき点をどうのように整理していくかを 4 つのプロセスに従って説明する。

（１）クラウドポートフォリオの作成

利用者の立場より、クラウドの利用が想定される例（ユースケース）と利用者におけるクラウドの使い分けの考え方について整理する。これにより、クラウド利用のメリット、デメリットを明確にすることができる。

（２）クラウド利用の課題、問題点の整理

クラウドサービスのより具体的な利用について、SLA や制度上の観点、及び、事業者やサービス選択の留意点、更にクラウドサービス調達時の留意点について記述する。これにより、利用時の課題、問題が事前に明確になり、対策を立てることができ、利用者とサービスの合意ができることにより、サービス時のリスクを削減することができる。

（３）クラウド構築の課題、問題点の整理

プライベートクラウドを中心に自組織内にクラウドを構築する際のスコープ、メリットから必要となる機能要素、構成、運用に至るまでの留意点を整理する。この整理により、構築時のリスクをクラウドセンタと共有することにより、軽減することができる。

（４）仮想化技術の課題、問題点の整理

クラウドサービスで重要な役割を果たす仮想化技術について、期待される効果、サーバ、ストレージ、ネットワーク、デスクトップにおける仮想化の考え方、仮想化技術の実装時の留意点等、仮想化技術の概観を整理する。この整理により、仮想化によるリスクをクラウドセンタと共有することにより、軽減することができる。

■本手法に関連する対策事例

「教訓Ｇ７」「教訓Ｇ９」の対策

２．５　障害管理の取組み

IPA では、IT の信頼性・安全向上の観点から、情報システム の組織的マネジメントやシステム障害事例収集・分析に取り組んできた。その取組みの一貫として、「障害管理の取組みに関する調査」報告書（2012 年）をまとめ、発表した（文献 4)。

本調査では、 重要インフラ分野における情報システム運用・障害管理の先進的共通な取組みやプロセスを抽出し、これらを包含する体系的枠組みとして、国際標準規格に準拠した「障害管理フレームワーク」を策定し、自社の情報システム運用・障害管理の取組みを点検する際に参照できるガイドとしてとりまとめた。要点は、以下の通りである。

（１）情報システム障害管理における組織・人材の課題

過去の大規模な情報システムの障害事例を分析した結果、システム障害防止のためには、以下の共通的課題への対処が必要である。

・経営層による情報システム運用への関与
・現場への体系的な障害管理マネジメント体制の構築
・障害管理マネジメントを統括する責任者の配置

（２）障害管理に求められる組織のあり方

情報システムの運用品質の継続的な向上に取り組むための組織・プロセスは、国際標準規格（ISO/IEC38500、ISO/IEC20000）に準拠し、経営層による「IT ガバナンス」と、情報システムを運用する現場の「IT サービスマネジメント」の両方の観点を踏まえ、組織化することが必要となる。また、情報システムの運用現場では、多様な職種の要員が役割を分担し業務に従事しているため、円滑なコミュニケーションとチームワークを促進する等、適切なマネジメントを行う「情報システム管理責任者」の配置が重要である。

（３）情報システム管理責任者のミッションとスキル

情報システム管理責任者は、経営層が決定した「情報システムの信頼性方針」を現場で実施し、情報システムの適切な運用管理（IT サービスマネジメント）を通じて、顧客に価値と満足を提供することに責任を持つ。情報システムの安定稼働を維持し、顧客に新しい価値や満足を提供することへの使命感を自覚することは、情報システム運用管理チームを統率するリーダーシップを発揮するための原動力となる。また、情報システム管理責任者に求められるスキルには、危機管理能力、クリティカルシンキング・問題発見力、コミュニケーション能力等他分野のマネジメントにも共通して必要となるスキルに加えて、情報システムに関する専門知識・経験等が挙げられる。

（４）情報システムの「障害管理フレームワーク」

「障害管理フレームワーク」は、国際標準規格（ISO/IEC 38500、ISO/IEC20000）に準拠し、経営層（トップマネジメントと CIO による「IT ガバナンス」と、運用現場の情報システム管理責任者による「IT サービスマネジメント」を構成する運用・管理プロセスを定義し、経営層と現場の連携のあり方と、各プロセス間の相互関係を明確化したもので、以下 3 つのパートで構成されている。 このフレームワー

クを利用することにより、自社の現状の取組みを点検・改善することが可能となる（図２．５－１）。

① 経営層（トップマネジメントと CIO）が関与するガバナンスのための 3 つのプロセス

・信頼性方針の策定、障害管理目標の達成判断、情報システムの運用状況の監視

② 現場の情報システム管理責任者が情報システムの高信頼化や安定運用を維持するマネジメントのための 6 つのプロセス

・障害管理目標の設定、運用、障害対応、再発防止、障害記録の確認、障害の予防・プロセス改善

③ ガバナンスとマネジメントで構成される障害管理の継続的改善のための 2 つのプロセス

・障害管理体制の構築、障害管理活動の有効性を高める方策

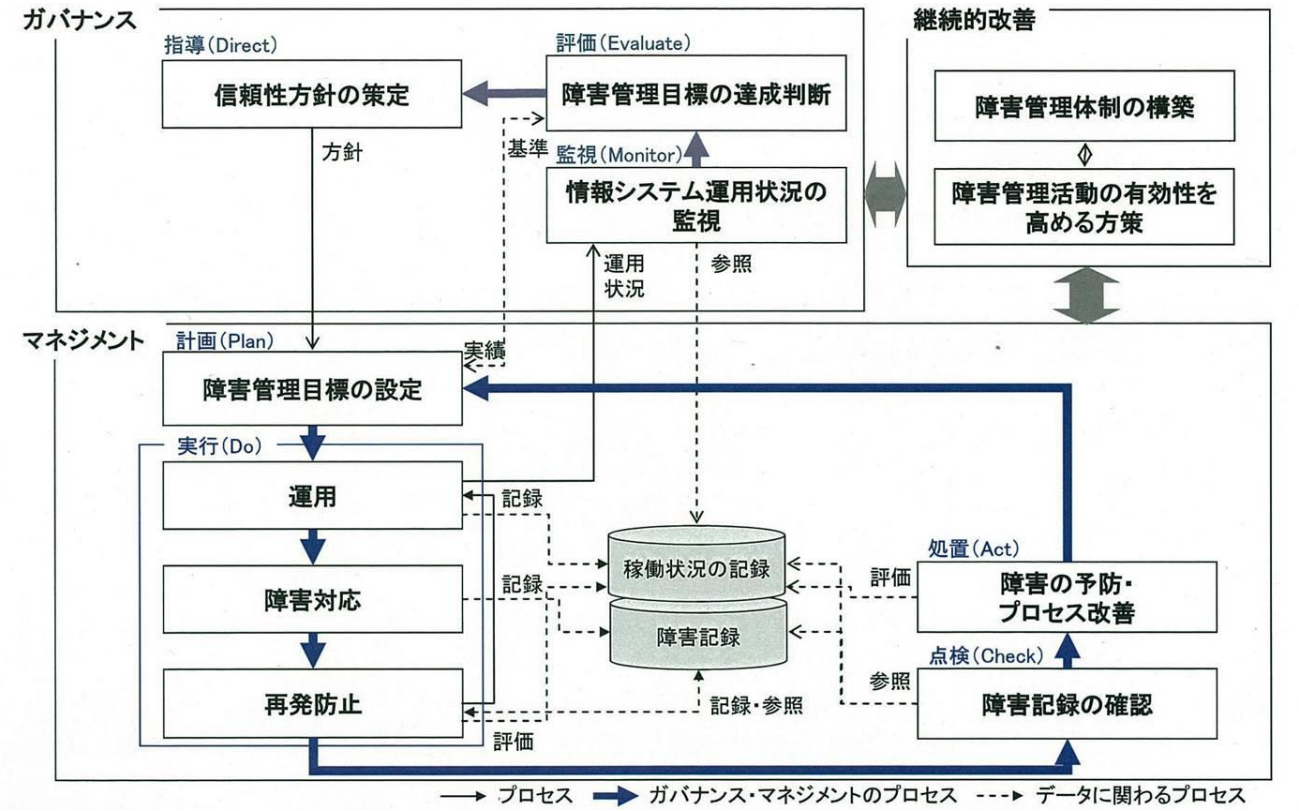


図２．５－１　障害管理フレームワーク[2]

■本手法に関連する対策事例

「教訓Ｇ４」「教訓Ｇ６」「教訓Ｇ７」「教訓Ｇ８」「教訓Ｇ９」「教訓Ｇ１０」「教訓Ｇ１１」「教訓Ｇ１７」「教訓Ｔ１５」「教訓Ｔ３１」の対策

[22] IPA『「障害管理の取組みに関する調査」調査報告書』2012 年　p.34

２．６　障害再発防止のための組織的マネジメント

IPA は、情報システムの管理体制を整備して障害の発生や影響が事業者内外に及ぶことを抑えている事業者の事例を調査し、その取り組みに共通する構造および特徴的な事例を整理し「情報システム障害の再発防止のための組織的マネジメントの調査 WG 報告書」としてまとめた（文献 5)。

調査の結果、障害の低減に成功している事業者は、品質向上の取り組みを PDCA サイクルで実践したり、稼働品質の目標達成に責任を負う管理責任者が PDCA サイクルを回したりしていることがわかった。

本報告書では、情報システム部門の品質管理責任者（運用部門における責任者など）または事業者の情報システム全体に責任を持つ CIO などが、事業者内の取り組みを他事業者と比較したり、どのような取り組みが必要かについて事業者内で対話を行う上での参考情報を提供したりしている。

以下に、対策手法事例として活用できるポイントを述べる。

（１） 品質目標に対する品質向上施策を管理し「稼働品質」を向上

情報システムの「稼働品質」が安定している事業者とは、品質目標に対して、品質向上施策を実行することにより、「稼働品質」が十分達成されている事業者と言える。

今回の調査対象の事業者は、この状態を前提としつつ、以下の点をチェックし、情報システムの「稼働品質」の維持・向上に必要な活動の展開を継続していた（図２．６－１)。

① 情報システムの中身はどのように変わったか？ 外部環境（利用者の層や利用の仕方を含む）にどのような変化が生じているか？ それによって品質向上施策の手直しは必要になっていないか？

② 今年度の「稼働品質」の目標を達成するためには、どのような品質向上施策の手直しが必要になるか？

③ これまでの「稼働品質」の実績と品質向上施策との関係で見逃されていることはないか？

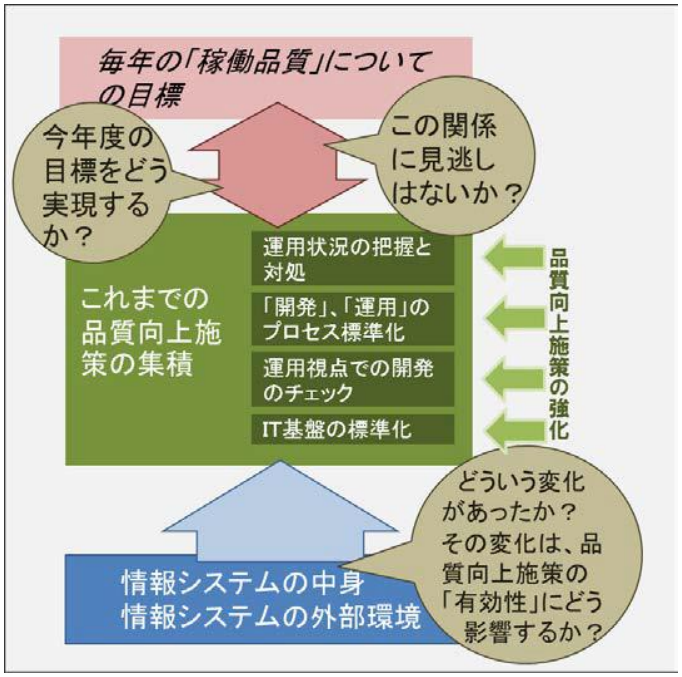


図２．６－１　「稼働品質」の目標と品質向上策の関係の管理[3]

[3] IPA『情報システム障害の再発防止のための組織的マネジメントの調査WG報告書』2012 年　p.27

（２）「稼働品質」についてのリスクの共有

情報システムに「稼働品質」の目標を立てそれを達成するための様々な管理がうまく行っているか、情報システムの「稼働品質」に危うい点があるとすればそれはどこか、ということについて事業者内で判断し、状況共有する仕組みの事例として金融事業者によるリスク管理の方式の概要を紹介する（図２．６－２）。

この金融事業者で行われていることを列記すると、以下になる。

① 情報システムの「稼働品質」に関するリスクについて、「リスク管理の要点」の説明や、リスクの識別の仕方がドキュメント化され、情報システム部門の要員全員に提示されている。

② 個別の情報システム部門を管理する要員から、情報システム全体の「稼働品質」の管理責任者へのリスクの報告方式（タイミング、報告の書式、報告先）が定型化されている。

③ 情報システム部門の要員や「稼働品質」についてリスクを識別する基準情報は、個別の情報システムのプロファイル情報としてまとめられている。このプロファイル情報は常に現状を反映したものになるよう更新されている。

④ 「稼働品質」のリスク情報は、情報システム部門の上位層に向かって情報集約されていき、毎月１回の会議にて情報システム部門のトップ（経営層）に報告され、評価を受ける。

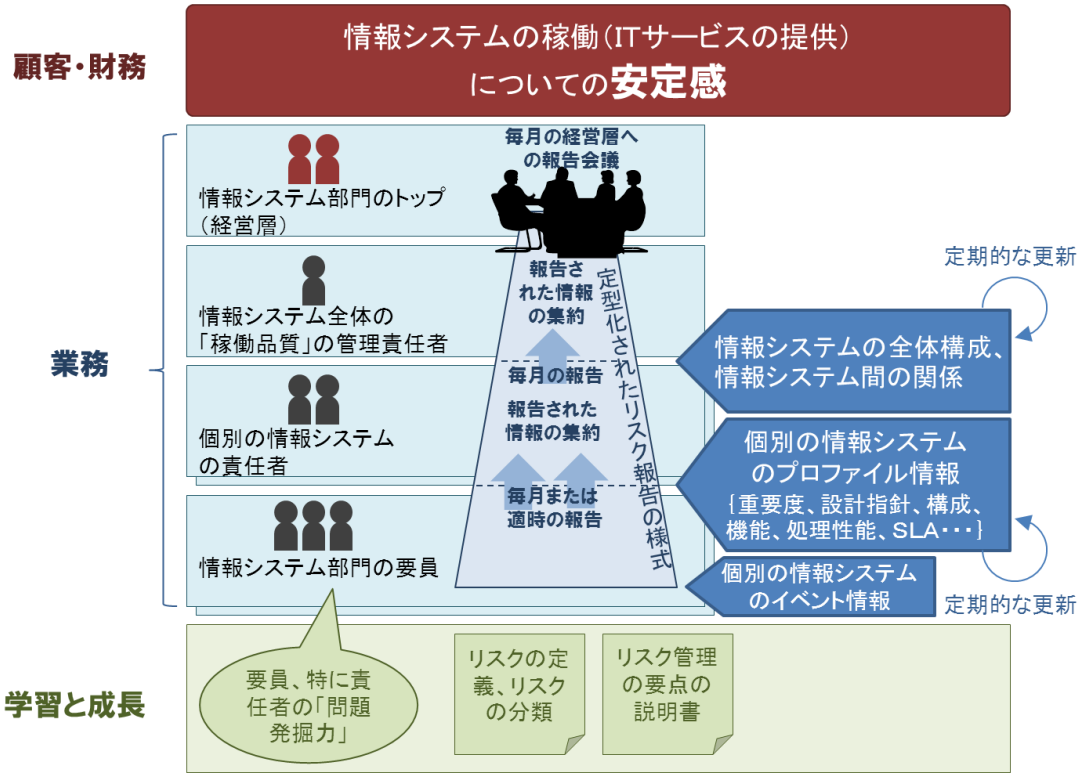


図２．６－２　情報システムの「稼働品質」についてのリスク管理の構造（例）[4]

■本手法に関連する対策事例

「教訓Ｇ１２」「教訓Ｇ１３」「教訓Ｇ１４」「教訓Ｇ２０」「教訓Ｔ３１」の対策

[4] IPA『情報システム障害の再発防止のための組織的マネジメントの調査WG報告書』2012 年　p.34

２．７　問題解決プロセス

IPA では、ソフトウェア開発の品質向上を目的として開発・運用プロセスをまとめた「共通フレーム」を作り、随時改訂してきたが、最終的な運用、つまりシステムの利用を想定した要求品質が曖昧であるという問題が生じ、2013 年 3 月公開の「共通フレーム 2013」では、運用のプロセスを大きく見直している。

そこでは、業務運営と IT の関係の全体像を明確にし、「ソフトウェア」「システム」に加えて、「業務」「経営（ビジネス)」を併せた四つの関係を模式化して、「運用を見据えた開発」のプロセスを示した。既存のシステムを含め、企業全体の業務運営の視点で運用を考えることを不可欠とした。

その中で、問題解決のプロセスが加わった。問題解決には、システムの要件に問題があれば要件定義からやり直す、システム構成が問題なら方式設計からやり直す、といったプロセスがある。共通化フレーム 2013 では、運用側からこの問題解決プロセスにつなぐプロセスを定義した。

これらのプロセスの全体を通じて、運用から始まって、業務改革、開発、そしてまた運用に戻るという継続的なサイクルを実現する。

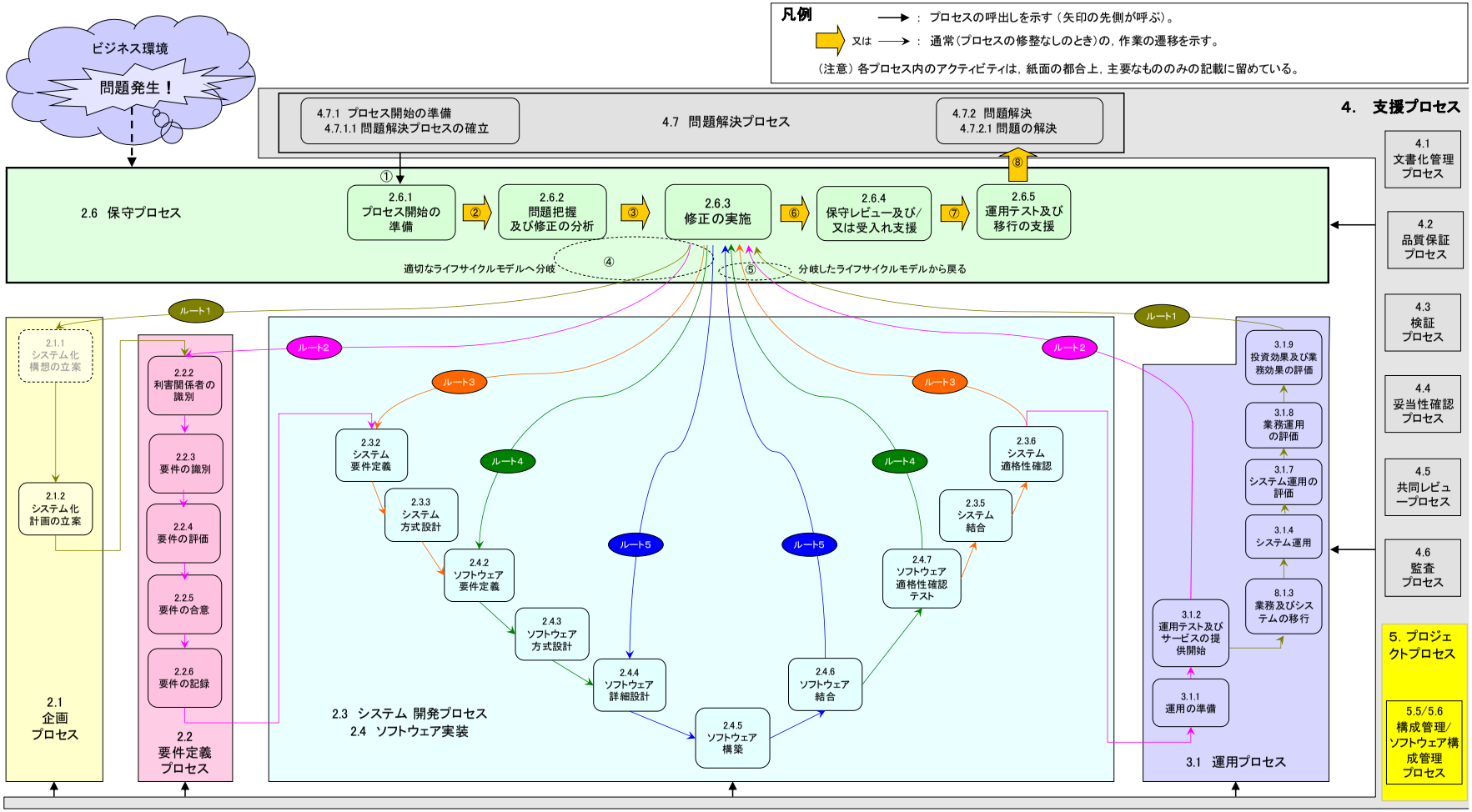
運用を起点としたこのサイクルでは、ユーザ企業側のオーナーシップがより重要になる。経営者も業務部門もシステム部門も深く関わって、業務運営の部分まで含めてオーナーシップを発揮しなければならない。

次頁に、「共通フレーム 2013」から、「図表４－２１　保守プロセスを中心とした、問題解決に関係するプロセス関係図」[5]を紹介する。

■本手法に関連する対策事例

「教訓Ｇ１０」「教訓Ｇ１１」「教訓Ｇ１５」「教訓Ｔ２５」の対策

[5] IPA『共通フレーム 2013
～経営者、業務部門とともに取組む「使える」システムの実現～』2013 年　巻末折込



ルート1 ： 現行事業の大幅な見直し等に伴い，システムの新規取得または現行システムの改良について検討する場合で，「2.1 企画プロセス」⇒「2.2 要件定義プロセス」⇒「2.3システム 開発プロセス／2.4ソフトウェア実装プロセス」⇒「3.1.2 運用テスト及びサービスの提供開始」⇒「3.1.7 システム運用の評価」⇒「3.1.8 業務運用の評価」⇒「3.1.9 投資効果及び業務効果の評価」⇒「2.6.3 修正の実施」を辿るルート。 例えば，新規業種を現行事業に吸収統合，現行事業の抜本的な効率化のための見直し等を検討する場合。

ルート2 ： 新しい業務や運用のあり方から検討する場合で，「2.2 要件定義プロセス」⇒「2.3システム 開発プロセス／2.4ソフトウェア実装プロセス」⇒「3.1.2 運用テスト」⇒「2.6.3 修正の実施」を辿るルート。
例えば，業務内容（業務フロー，入出力情報，組織の責任・権限）の追加・変更，業務特性（ルール，制約）の追加・変更，大幅な運用要件の変更を検討する場合。

ルート3 ： ハードウェア，基盤ソフトウェア（OS，ミドルウェア等），アプリケーションソフトウェアなどシステム全体の最適化や変更を検討する場合で，「2.3.2 システム要件定義」⇒「2.3.3 ～ 2.3.6の各アクティビティ」⇒「2.3.6 システム適格性確認テスト」⇒「2.6.3 修正の実施」を辿るルート。 例えば，明確化された業務要件のシステム化検討，ハードウェアが追加されるなどシステム化方式やシステム間インタフェースを追加検討する場合。

ルート4 ： 基盤ソフトウェア（OS，ミドルウェア等）や，アプリケーションソフトウェア等のソフトウェア要件の追加・変更を検討する場合で，「2.4.3 ソフトウェア要件定義」⇒「2.4.4 ～ 2.4.6の各アクティビティ」⇒「2.4.7 ソフトウェア適格性確認テスト」⇒「2.6.3 修正の実施」を辿るルート。 例えば，アプリケーションソフトウェア仕様（画面，バッチ，帳票，DB，ファイル等）が変更になる場合，OSや通信ミドルウェアやセキュリティソフトウェアを更新する場合。

ルート5 ： コーディング，コンパイル＆テストを実施するソフトウェアユニットレベルの変更が必要な場合で，「2.4.4 ソフトウェア詳細設計」⇒「2.4.5 ソフトウェア構築」⇒「2.4.6 ソフトウェア結合」⇒ 「2.6.3 修正の実施」を辿るルート。
例えば，プログラムレベルの仕様追加やバグ修正の場合。

図4-21 保守プロセスを中心とした，問題解決に関係するプロセス連関図

２．８　プロセス改善

発生したシステム障害を教訓として予防対策を行う上で、ガバナンス／マネジメント領域で実行しなければならないことは、作業ミス、要件の確認漏れ、やるべきことを抜かしてしまった、等の作業プロセスの問題で起きたシステム障害を無くすことである。

ソフトウェアは人が作るものであり、人が運用保守していくものである。システム開発者、運用担当者個人の方法や技量だけに頼って役割や責任が不明確なまま作業を進めてしまうと、属人的な仕事のやり方（ひとり親方）になってしまい、以下の事態を引き起こすことになる。

・製品（サービス）の品質（Quality）が、安定しない。
・納期（Deliver）が、不確実（納期を必ずしも守れない）になる。
・結果的に、コスト（Cost）が増大する。

組織が、顧客に提供する製品（サービス）品質を維持し、または向上させていくためには、組織の活動基盤となる“プロセス品質”が重要である。しかし、これまでに多くの組織がプロセス改善に取り組んできたが、必ずしも成功していない事例が数多くある。

その要因としては、プロセス改善に関係するメンバの認識が合っておらず、メンバの合意も得られていない状態で強引に進め、広く普及している「成功事例」に自組織を形式的、表面的に合わせて対応してしまったこと等が挙げられる。

このような要因への対策手法として、「プロセス改善ナビゲーションガイド」（文献 7）で述べている、「プロセス改善における 10 の勘所」（図２．８－１）と「プロセス改善８ステップ」（図２．８－２）を紹介する。

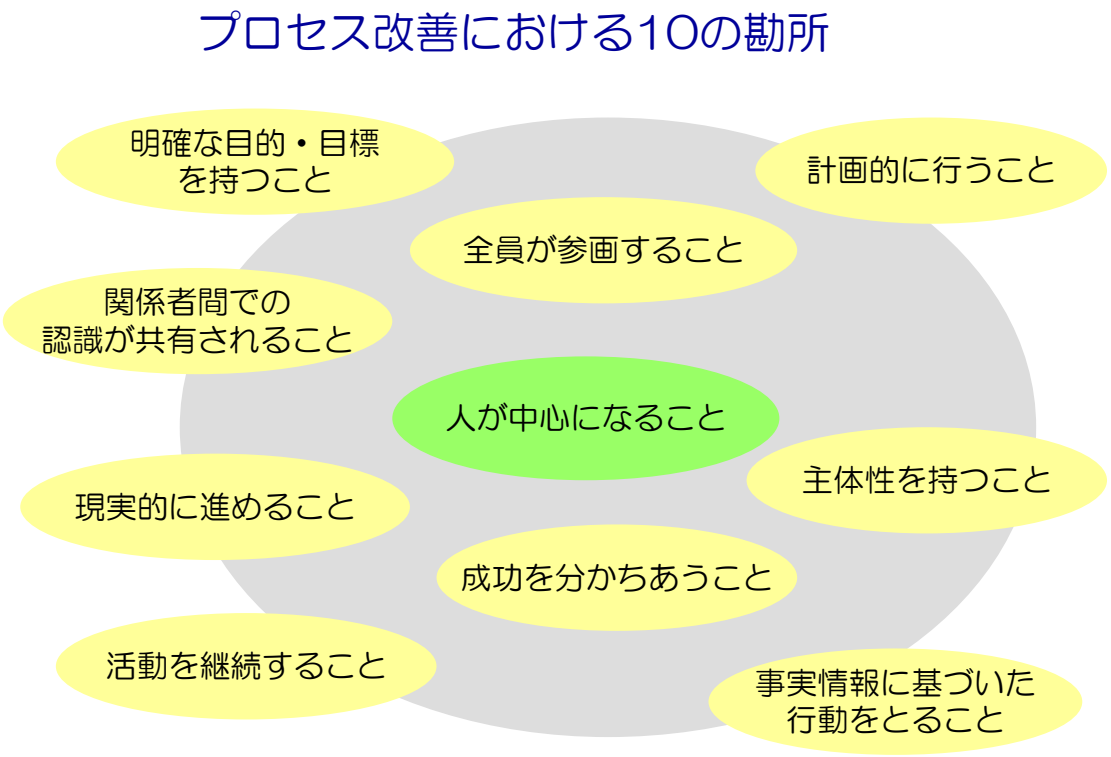


図２．８－１　プロセス改善における 10 の勘所[6]

[6] SEC BOOKS プロセス改善ナビゲーションガイド<虎の巻編> の図の簡略版

「プロセス改善における 10 の勘所」では、プロセス改善を成功に導くための要点、取組みで忘れてはならない勘所を 10 点に整理して述べている。図では 10 個のタイトルしか表示していないが、参考文献の本文には、詳しい説明があるので、参照願いたい。

「プロセス改善 8 ステップ」は、以下の手順で進める。

・“良い”と信じる方向を「見つけ出す」

・“良い”と信じる方向に自分たちが向かっていることを「認識する」

・思った方向に向かっていないことが分かったら軌道を「修正する」

このような一連の活動を実施しながら、自分たちの「仕事」全体を見直していく「改善活動」を進めることになる。

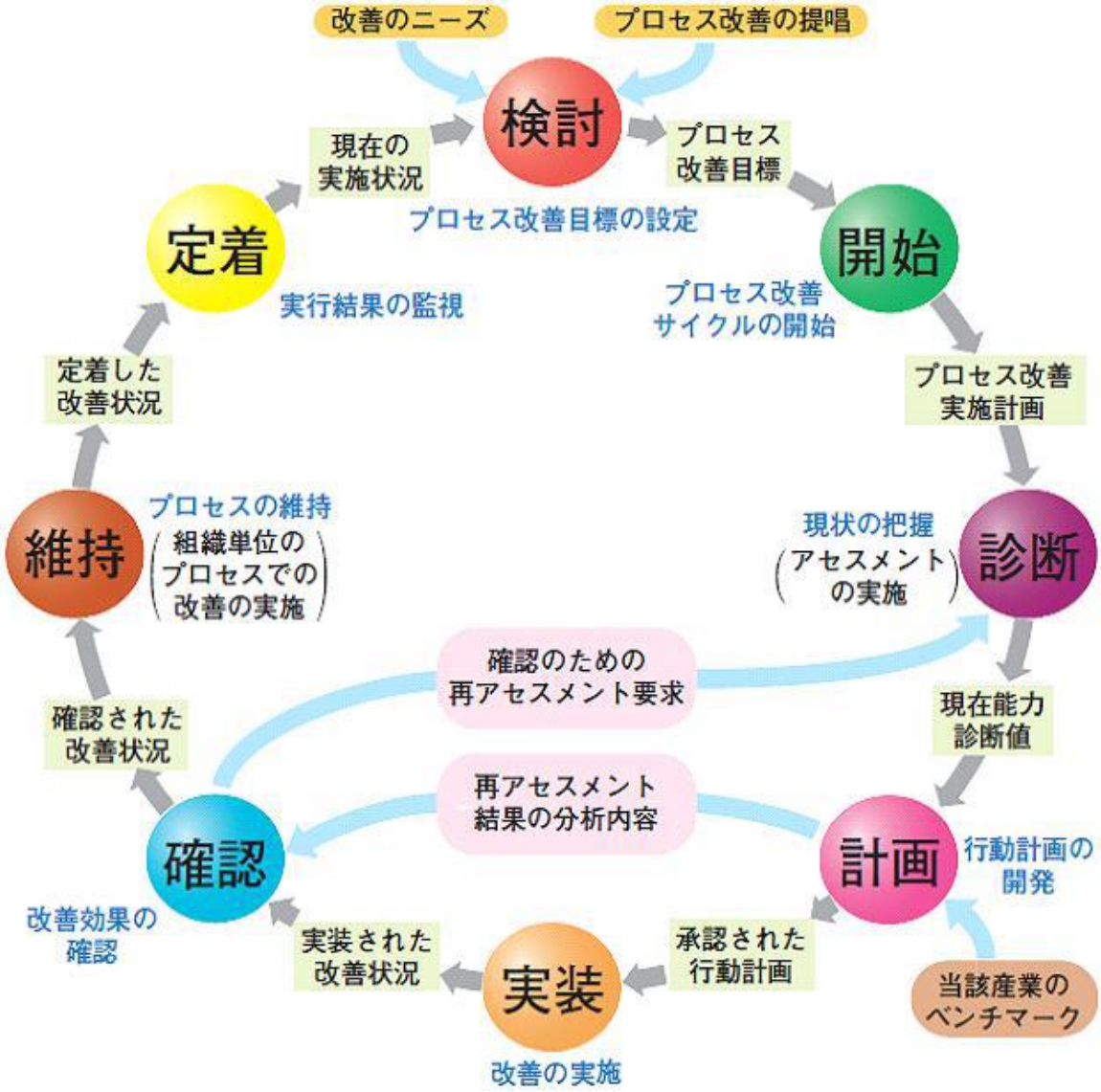


図２．８－２　プロセス改善 8 ステップ
（15504-4 での改善サイクル）[7]

[7] SEC BOOKS プロセス改善ナビゲーションガイド<虎の巻編> 図 1-1

「プロセス改善ナビゲーションガイド」では、この「プロセス改善８ステップ」を、執筆陣の実務経験に基づいて、プロセス改善を始めることになった方々が遭遇した数々の壁（課題）を題材として取り上げ、それらの解決のためのヒントとして解説しているので、参照願いたい。

尚、８ステップとは、図２．８－２の

検討→開始→診断→計画→実装→確認→維持→定着　を指す。

■本手法に関連する対策事例

「教訓Ｇ４」「教訓Ｇ５」「教訓Ｇ６」「教訓Ｇ９」「教訓Ｇ１９」「教訓Ｔ１７」「教訓Ｔ２１」の対策

２．９　運用時の定量的信頼性向上方法

システムの信頼性向上のためには構築時だけではなく運用時の取組みも重要である。システム構築時についてはかなり前から定量的管理の手法等がまとめられ、事例の実績も多い。しかし、運用時についてはこれまで十分整理されていない。そこで IPA では、主として定量的なアプローチによる運用時の信頼性向上方法（使用指標、指標測定データに基づく対策手法、予兆などの観測項目、観測データの分析手法など）に対する取組みの現状を明らかにし、その課題を見出すことを目的として調査を実施し、その結果を「情報システム運用時の定量的信頼性向上方法に関する調査報告書」（文献 8）として取りまとめ、公開した。

本調査報告書では、運用に関わる指標、プロセス、ツールなどについて定量的管理の視点から、文献や公開情報の収集及び各産業分野の企業などへのヒアリングを通じて調査を行い、その結果を次の項目について取りまとめている。

・現在の IT システムを取り巻く環境やシステムの特徴
・運用プロセスとして標準的に参照されている国際規格
・運用時の定量的指標として提案されたり実際に使われたりしている指標の事例と考察
・運用時のプロセス管理や障害予知に関するツール及び研究事例
・運用に関する技術とツールの現状及び動向
・運用のプロセスや指標に関する実態及び運用時の信頼性に関わる知見や問題点
・運用時の信頼性向上の観点での現状分析及び今後の動向と課題
・公的機関などにおける今後の取組み課題の提案
・IT システムのライフサイクルにおける運用の位置付けと運用時の信頼性向上に関する方向性

この調査の結果、下記の現状や課題が明らかとなった。

多くの組織において

･IT システムの運用ではなくサービスの運用としての視点を持っている
･ITIL を参考にして独自に運用プロセスを作成している
･KPI を設定して運用管理を行っているが、SLA と KPI の関連付けは課題
･統合監視ツールを活用しているが、障害予兆検知は今後の課題
･運用の人材育成やスキルの評価については問題意識を持っている

■本手法に関連する対策事例

「教訓Ｇ１５」「教訓Ｇ１６」「教訓Ｇ１８」「教訓Ｔ２３」の対策

３．技術領域

技術領域の教訓における対策の手法と事例を説明する。

技術分野の領域は非常に幅広く、また詳細に亘るため、この対策集でひとつひとつの文献を詳細に述べることは不可能である。よって、ここに記述されていることは、概要紹介にすぎないため、直接参考文献を当たっていただきたい。

参考文献が示されていないところは、この教訓集で初めて取り上げた対策になるので、直接、教訓集を読んでいただきたい。これらについては、今後、広く該当する文献を求めていきたい。

３．１　トレーサビリティ管理

一般的には、ソフトウェアのトレーサビリティ（traceability、追跡可能性）として、以下の 2 つが挙げられる（文献 10)。

（１）製品に関するトレーサビリティ

（２）文書、及びデータに関するトレーサビリティ

（１）は、ソフトウェアアイテム（ソフトウェアの部品の最小構成要素）が、製品にどのように組み込まれているかが追跡できることであり、ハードウェア製品等の部品が、どのように製品に組み込まれているかを管理できることである。

（２）は、ソフトウェア要件が、製品のどの部分で実現されているかを追跡できることを意味し、逆にできあがっているソフトウェアの構成部品（例えば、ある実行モジュール）が、どのソフトウェア要件を実現するためのものかを追跡できることも意味する。

３．１．１　製品に関するトレーサビリティ

製品に関するトレーサビリティは、ISO9001（品質マネジメントシステム要求事項）で定義されている（文献 9)。

・組織は、製品について、どの程度の詳しさで、トレーサビリティが行えるようにするのかを決める。
・組織は、トレーサビリティを行うために、製品に一意の識別をつけて管理する。
・その番号を記録に書き込んで、どの番号の製品をいつ、誰が、どのように処理したかが、分かるようにする。

さらに補足説明として、情報システム開発等の分野では、構成管理（構成部品ごとに、変更やその影響を管理するシステム）を行い、製品毎のトレーサビリティに必要な情報を記録するとしている。

■本手法に関連する対策事例

「教訓Ｔ１２」「教訓Ｔ２０」「教訓Ｔ２７」の対策

３．１．２　文書及びデータに関するトレーサビリティ

信頼性の高いソフトウェアを作るためには、ソフトウェアの開発フェーズの各段階において、前フェーズでの欠陥を作り込むことを予防することが重要であり、そのためのツールとしてトレーサビリティ管理を活用することができる（文献 10)。

トレーサビリティの高いシステムでは、ソフトウェアの要件定義書、仕様書、変更履歴、テストや障害の記録、ソースコード、バージョン、実装等を相互に参照・確認できるような仕組み、紐付けることができる。つまり、トレーサビリティを実現するには、ソフトウェアの様々な構成品に対して、自分の変更が他に影響を与える場合と、他に与えた影響がどのように自分に帰ってくるか、といったソフトウェア構成品間の関係（トレース情報）を記録して、お互いに矛盾のない関係を管理する、構成管理を築くことにある。

■本手法に関連する対策事例

「教訓Ｔ５」「教訓Ｔ１４」「教訓Ｔ１８」の対策

３．２　「見える化」手法

IT プロジェクトにおいては、目標と計画をきちんと立て、計画との差異を把握しながらシステム開発を進めることが重要である。そのためには、実態（計画との差異）がステークホルダに見えなければならない。実態が見えなければ、状況を理解することはできないし、プロジェクトに潜んでいるリスクを見つけることができない。

「見える化」手法は、そのような IT プロジェクトのあらゆる現象（プロジェクト構成、要件定義、進捗、・・・）をステークホルダに「見える」ようにする手法である（文献 11)。

３．２．１　暗黙知の整備・有効活用

プロジェクトの状態を把握するためには、KKD（勘と経験と度胸）だけでなく、定性的、定量的なアプローチが必要である。

過去からのシステム要件やテストパターン等を暗黙知でなく形式知として整備し、改修や新たに再構築したりするときに、その形式知を鏡として検討することにより、要件の抜け漏れを防止し障害を回避する。

要件に記載が漏れやすい内容（業界常識、顧客常識、標準となっている業務手順、規約等）や、現場で確認しなくては判らない事象は、有識者による機能確認を行っても、見落とされる要件であり、「機能漏れ」となる障害を招く。

まず、実業務等の観察、インタビューやアンケートによる聞き取りを行い、暗黙知を形式知化すべき情報を収集する。

次に、それらの情報を要件として整理蓄積して管理（知識データベース等のツールを作成）し、そこに仕様変更、機能追加等の新たな要件を追加登録（記録）していく。これにより、暗黙知が形式知化される。

この管理された要件の記録を参照すると、設計時の考慮漏れやリスクの洗出しができる。

更に、それらの暗黙知から形式知化された要件は、第三者診断体制を組織化し、妥当性を検証する。

■本手法に対する対策事例

「教訓Ｔ３」「教訓Ｔ２０」の対策

３．２．２　俯瞰図

障害対策は、障害を起こした箇所だけの対策を考えがちであるが、システム全体を俯瞰する対策を活用することで障害発生時の復旧時間の短縮が可能であり、システムの安定稼働（障害発生頻度の減少）が保たれる。

そのためのツールのひとつとして、プロジェクトを全体的にとらえるための「俯瞰図」がある。

システム障害対策では、「障害箇所だけ見て全体を見ず」という近視眼的な状態に陥り、結果的に対策が十分でない場合が生じる。プロジェクトのステークホルダが、客観的な視点で、「どう動けばよいか」、「どこに問題が生じるか」を知ることが重要である。そのためには、プロジェクトの全体像を高い位置から眺めた俯瞰図の作成が役立つ。

俯瞰図からの障害対策は、当該システム及び周辺システムの全体構成、関連を俯瞰的に表すことにより、改修時の影響範囲、インタフェースミス等を防止することができる。

また、俯瞰図によってシステム全体で対策を検討することができ、センタ（オペレータ、司令室等）で対処できる機能を考慮することができる。(例えば、司令室からの障害機器の切離し、切替え。オペレータによる機能の停止、再開、入力制限等)

■本手法に対する対策事例

「教訓Ｔ２」の対策

３．３　要求獲得手法

要件漏れによる変更管理が行われず、システム障害となることがある。このようなシステムの変更を行わなくてはならない時点を見逃がさないため、要求工学による要求獲得プロセスを活用する（文献 12)。

システムの変化点、変更点の見落としを防ぐための手法としては、システムの環境条件の変化が重要なポイントである。

要求獲得プロセスは、８つのフェーズがある。そのうち、要件が変更された時に絞った「要件漏れ」を防ぐ４つのフェーズ（①～④）と、更に２つのフェーズ（⑤～⑥）を追加する（図３．３－１)。

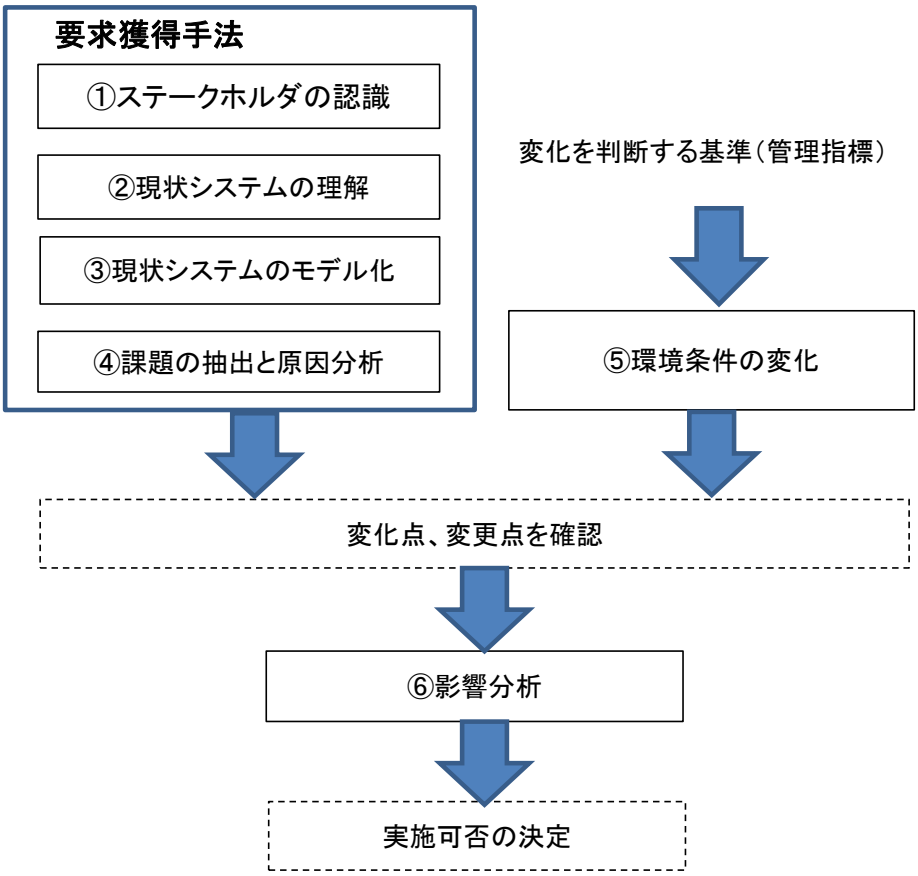


図３．３－１　要件確定の手順

① ステークホルダの識別

ステークホルダ（システムに関与する個人、グループ、組織等）は、システムに対する要求の源泉である。

② 現状システムの理解

シナリオを作成し、使用に関する具体的なストーリーを作る。ストーリーは情報システムに限定されず、目的を持って一定の規則に従って、ユーザの言葉でユーザの操作を記述する。

シナリオ発見の方法は、インタビュー、観察、要求ワークショップ等がある。

③　現状システムのモデル化

モデルを作成することにより、要求の漏れ、矛盾を発見することができる。

④　課題の抽出と原因分析

ステークホルダの中からキーパーソンを選び出し、現状の問題点を聞き出す。また、問題関連図等を作成し、問題の原因、影響を分析する。

⑤　環境条件の変化

システムが監視・制御する対象と仕様の変化点（管理指標）を明確にする。対象時間、対象機器の動き、機器数の変化等もシステムに対する環境条件の変化点と考える。

また、サービスの視点での変更点も重要な変化点（管理指標）である。

⑥　変更の影響分析

環境条件の変化が明確になった場合、その変更要件の影響分析を行う。併せて、リスク分析を行う。

この分析結果を審査し、変更要求としての実施可否を決定する。

■本手法に関連する対策事例

「教訓Ｔ４」「教訓Ｔ５」「教訓Ｔ２４」の対策

３．４　変更管理

「３．３　要求獲得手法」では、システムの変化点、変更点を捉え、その影響度がどの程度なのかを分析して、要件変更として確立するまでの対策手法であった。ここでは、要件変更を、実際にシステムへ反映するための管理手法「変更管理」について説明する。

変更管理は、要件変更作業を有効に行う手法である。また、「JIS Q 20000-1　情報技術　サービスマネジメント」においてもその管理手法について規定されている。（文献 13）以下、「JIS Q 20000-1　情報技術　サービスマネジメント」の変更管理について要約して説明する。

① 変更管理方針を確立し，以下を定義する。

（ａ）変更管理が制御している CI（注：configuration item 構成品目）

（ｂ）サービス又は顧客に重大な影響を及ぼす可能性のある変更を判断する基準

このうち（ｂ）は、「３．３　要求獲得手法」で導き出された環境条件の変化である「変化点」、「変更点」と密接に関連するため、十分検討する。

② 変更要求を記録し，分類し，評価し，承認する文書化された手順を作成する。

③ 全ての変更は，変更要求を用いて提起する。

④ 全ての変更要求は記録し，分類しておく。サービス又は顧客に重大な影響を及ぼす可能性があるとして分類された変更要求は，サービス変更の設計及び移行プロセスを用いて管理する。変更管理方針で定義された，CI に対する他の全ての変更要求は，変更管理プロセスを用いて管理する。

⑤ 変更要求は，変更管理プロセス及びその他のプロセスからの情報を用いて評価する。

⑥ サービス提供者及び利害関係者は，変更要求の受入れについて決定する。

⑦ 承認された変更は，必ず開発，試験のプロセスを実行する。

■本手法に関連する対策事例

「教訓Ｔ４」「教訓Ｔ５」「教訓Ｔ８」「教訓Ｔ１４」「教訓Ｔ１５」「教訓Ｔ２６」「教訓Ｔ２８」「教訓Ｔ２９」「教訓Ｔ３１」「教訓Ｔ３２」「教訓Ｔ３３」の対策

３．５　フェールソフト

フェールソフトとは、システムの障害時の際にも、正常な部分だけで稼働を継続させることを重視した考え方である。業務内容に基づいて、システム毎にポリシーを作成したうえで、フェールソフトを適用する。

具体的には、ハードウェア機器の故障、ソフトウェアの処理の異常等があった場合には、その部位を積極的に停止させ、システムから切り離す。場合によってはその系全体を放棄するといった考え方のもとに処理・対応する。

一方、そのような状況下で一部の部位や系をシステムから切り離しても、システム全体としてのサービスは継続できるように、フェールソフトの考え方に基づいて設計・運用する。

フェールソフトを実装しておくことで、重大障害が発生しても業務が継続可能となり、取引の停止や報道発表につながるような重大トラブルが未然に防止できるようになる。

■本手法に関連するする対策事例

「教訓Ｔ１」の対策

３．６　テスト技法

テスト環境と本番環境は同一が望ましいが、コスト面等からテスト環境が必要最低限の構成となることがある。その場合、シミュレーション手法、障害運用テスト等を効果的に行うためのツール（ハード／ソフト)、本番機の一時的利用等を検討・整備する。また、テスト不可能な場合は、その項目をリスクとして認識し、発生時のコンティンジェンシープランを整備し、CIO まで共有する。

３．６．１　テスト環境のリスク管理

テスト環境と本番環境の構成（ハードウェア、ソフトウェアや個々のソフトウェアのバージョン、パラメータ等）を極力同一にすべきであるが、合わせられない場合には、以下の対策を実施する。

- テスト環境と本番環境の差異を明確にする。特に、「ソフトウェア製品管理」、「ハードウェア製品管理」、「アプリケーション管理」等重要項目を全て洗い出した「差異分析」を行う。
- テスト仕様において、事前にテスト環境で確認できない項目、機能が存在する場合、事業部門／情シス部門開発担当／情シス部門運用担当の 3 者で、リスク分析を行う。
- そのリスク分析結果を基にそのリスクの度合いに応じて、リスク対策やコンティンジェンシープランを立案して、事業部門、または経営トップへリスクを伝える。
- 本番環境の保守作業のリスクをステークホルダ（経営トップ、製品ベンダを含め）で共有する体制を作る。
- 大きいリスクは、経営トップが判断する。

■本手法に関連する対策事例

「教訓Ｔ６」「教訓Ｔ９」の対策

３．６．２　シミュレーション手法

制御系システムでは、装置の動きの微妙なタイミングの試験をすべて実機で確認することは不可能であるため、シミュレータによるテスト環境を準備することが有効である。

また、対外システムとの連携等を行うシステムの機能テストを自社内で実施するためには、対外システムをテスト環境に持ち込まなくてはならず、それも不可能であるため、同様にシミュレータによるテストが有効である。

シミュレータの開発を行うためには、現実の制御装置なり、対外システムのプロセスを分かりやすく可視化し、プロセスの骨子を見極めて「モデル」化（モデリング）する。

■本手法に関連する対策事例

「教訓Ｔ３」の対策

３．６．３　テスト網羅性の高度化技法

従来のソフトウェアテストでは、テストに対する要求の分析が不十分なまま、いきなり仕様書からテストケースを作成してもテスト全体の網羅性を確保することは困難である。テストに対する要求の分析をきちんと実施するとともに、テスト設計における網羅性とピンポイント性についてよく理解し、それぞれ適切なテスト技法を使用することが重要である（文献 14)。以下、テスト網羅性の高度化技法について説明する。

（1）テスト要求分析

これまで多くのソフトウェアテストの現場では、仕様書から直接テストケースを作成し、続いてそのテストケースに与えるテストデータを用意し、テストを実施するという方法が採られてきた。ところがソフトウェアの大規模・複雑化にともない、この方法ではシステム全体のテスト網羅性が十分に確保できなくなってきた。そこで、ソフトウェアテストにおいてもテスト設計前にテスト要求分析を行う。

テスト要求分析では、始めにテスト対象の情報収集を実施する。情報にはお客様に属するものと、開発プロジェクト自体に属するものがある。

お客様に属する情報を収集する目的は、真の要求の把握を行うためである。ソフトウェアテストは、仕様通りに「正しく」製品を作っていることを検証することが第一だが、お客様の要求と合致しているか、つまり「正しい」商品を作っているか、お客様の問題や課題はこのソフトウェアで本当に解けるのかといった妥当性確認をするという目的もある。

次に、開発プロジェクトの情報を収集する目的は、開発プロジェクトの進捗状況、ソフトウェア開発力（個々人のスキルとチームスキル)、投入リソース（人・物・金)、完成度、ソフトウェアの流用元、選択技術（開発言語、OS、COTS、FLOSS 等々)、ここに障害があるとシステム全体が止まってしまうといったシステムの急所（Single Point of Failure）等々の情報である。

テスト要求分析では、こうして集まったテストベースを参照しながら、テスト要求分析手法等を使用してテスト対象を扱いやすい単位まで分解・整理しながら、テストの全体を明らかにする。

（2）テスト観点とテストアーキテクチャ設計

テストには、様々な「観点」が必要といわれている。しかし、これまで用いられてきた観点はテストとして使用するにはあまりに粒度が粗く、テスト設計を実施すると機能性に偏ったテストになりがちであった。テストとして重要なことは、テストの全体像をテスト要求分析で把握しながら、すべてのテスト観点を漏れなく、重複なく挙げることである。

このようなテスト観点を用いて、テストアーキテクチャ設計を行う。テストアーキテクチャ設計とは、テスト観点を様々な面として捉え、その面をどう組み立てて、テストの網羅性を設計していくかをモデルとして設計していくことである。

（3）高度化技法

テストアーキテクチャ設計によって作成したテスト構造には、そのままテストケースに変換できる単純なものもあれば、テスト技法を使用してテスト条件を明らかにした上で、テストケースを実装する必要があるものもある。そのために、網羅的な技法とピンポイント的な技法がある。

まず網羅的な技法の代表として、「直交表を活用した網羅的な組み合わせテスト」、ピンポイント的な技法の代表である「シナリオを用いた効果的なピンポイントテスト」がある。

このような、テスト網羅性の高度化技法を活用し、テスト漏れに因るシステム障害を起こさない対策を立てることができる。

■本手法に関連する対策事例

「教訓Ｔ９」「教訓Ｔ１３」「教訓Ｔ２８」の対策

３．７　可用性管理

重要インフラシステムはサービスを停止することが許されない要件を持つものが多い。そのため、システムの稼働率を最大に向上させる必要があり機器の冗長化、活性保守等の仕組みを構築しサービス提供を継続することになる。

３．７．１　システムの冗長化設計

耐障害性を向上させ、可用性を確保するために、システムを構成する機器、ネットワークは極力冗長化構成とする設計を行う。障害の同時発生（二重障害、三重障害…）をどこまで想定して冗長化するかは、リスクとコストとの見合い（投資対効果）となる。

冗長化の方式は、稼働系と待機系（バックアップ系）をクラスタ構成で相互に死活監視するバックアップ方式と複数の機器で分散処理を行い１台の障害時に他の機器で縮退運転する並行稼働方式がある。

切替え方式では、稼働している機器は１台という前提があり２台が同時に稼働しているような状態にさせない工夫が必要となる。

■本手法に関連する対策事例

「教訓Ｔ１」「教訓Ｔ２」「教訓Ｔ７」「教訓Ｔ９」「教訓Ｔ１０」「教訓Ｔ１６」「教訓Ｔ３０」の対策

３．７．２　シングルポイントの洗出し

シングルポイントとは、冗長化している機器の中で、シングル（単一）構成となっている箇所を言う。

機器の冗長化を実施しても、それを制御する機構がシングルであると、その箇所で障害が発生すると継続サービスが不可となる。このような、シングル構成となっている機器をすべて洗い出し、その部品を準備し、対応手順等を用意する等の工夫が必要となる。

■本手法に関連する対策事例

「教訓Ｔ７」「教訓Ｔ１２」「教訓Ｔ３０」の対策

３．７．３　障害運用マニュアルの整備と訓練

冗長化構成としている機器は稼働系のダウン時に自動的に待機系に移行するための制御ソフトを備えているが、実際にそうなるかを全てテストする必要がある。また、ダウンまではしないが処理が異常に遅くなる（スローダウン）するような場合は、手動で待機系に切り替えることになる。手動切替え手順等の運用マニュアルを整備し、実環境で運用訓練を行っておく必要がある。

■本手法に関連する対策事例

「教訓Ｔ７」「教訓Ｔ１２」「教訓Ｔ１７」「教訓Ｔ３１」の対策

３．７．４　可用性の注意点：フルメッシュ構成

一般に、可用性を高めるためには、冗長化し、その上で、結合する機器同士をフルメッシュの構成にすることが、最善と考えていた技術者は、多いと思われる。

しかし、必要のないところまで、機器同士を結合させることは、一旦障害が起きた時に障害の影響範囲を広げてしまうことになる。

システムの特性を考えて、システム構成を決め、障害の影響範囲を押さえる対策は、可用性対策のポイントのひとつである。

■本手法に関連する対策事例

「教訓Ｔ１０」の対策

３．８　非機能要求グレード

情報システムに対する要求には大きく分けて 2 つ存在する。

ひとつは、業務実現に関する要求で、業務の機能そのものを示すことから「機能要求」と呼ばれる。もうひとつは、「非機能要求」と呼ばれる要求で、システム基盤に関する要求は、主にこの「非機能要求」である。非機能要求項目としては、可用性、性能・拡張性、運用・保守性、移行性、セキュリティ、システム環境・エコロジー等がある。

システム基盤に関する非機能要求を漏れなく明確化し、ユーザ／ベンダ間で認識を共有化することで、適切な情報システムを構築し、安定的なサービスを提供できるようにするためのツールとして、非機能要求グレードがある。（文献 15）

性能問題によるシステム障害の対策は、性能要件に基づく必要機器リソースの算出根拠を明確化し、要件の変化に対するリソース増強の手順を明確にしておくことである。そして、日々のデータ量の増減と機器リソースの利用率を監視し、閾値越えの場合に事前に策定した対応を行うことで未然に障害発生を回避することである。

性能・拡張性に関する非機能要求を合意する場合、「業務処理量」をはじめに決めることが一般的である。業務処理量があって、その上で性能要求を決める。「性能目標値」は、前提となる業務処理量を考慮し、システムの処理形態やピーク特性や縮退時の性能目標の設定を行う。

ピーク時の性能目標を決めるためには、処理毎に、通常時と比べた処理量がどの程度増えるのか、ピークとなる頻度や時間帯等が予見できるのか等を考慮する。縮退時の性能目標を決めるためには、処理毎に、優先度や重要度を確認しておく。

以下に、非機能要件での考慮すべき観点を示す。

・システム資源（CPU、ディスク等）
・特定業務のトランザクションの一時的な増加による影響
・各種運用管理ソフト（SNMP による機器監視ソフト等）の利用
・バックアップ切替え、縮退運転時での必要性能要件

等

■本手法に関連する対策事例

「教訓Ｇ５」「教訓Ｔ７」「教訓Ｔ８」「教訓Ｔ１１」「教訓Ｔ１５」「教訓Ｔ１８」「教訓Ｔ２２」の対策

３．９　仮想化技術

政府機関が行うITシステムの調達は、高度なIT知識が必要であるため、IPAでは、調達に必要な技術情報をまとめた「技術参照モデル（TRM：Technical Reference Model)」を提供し、政府の指針に従った公平なITシステムの調達を支援した。その内容は、典型的なシステム構成モデルや、特定企業に依存しない機能要件/非機能要件の記述例になっている。

その中で、クラウドサービス編の仮想化構築時の留意点が、一般の仮想化技術導入にも役立つと思われるので、以下に紹介する（文献16)。

・従来の各ハードウェア資源単位でのサイジングに対し、仮想リソースの割当単位、業務システムの単位、物理リソースの単位、リソースプールの単位に分けて検討する。
・従来ハードウェア層で、1:1若しくはN:1等のモデルで設計していたサーバの高可用性要件について、他の高可用性要件を考慮したうえで、仮想化層において、リソースプール全体でのN:Mモデルの適用を検討する。
・ハイパーバイザ、仮想サーバ、クラウド管理層の監視とセキュリティ、複数の仮想資源の構成管理は、従来のシステム設計に追加して検討する。
・仮想化によるパフォーマンスの低下、リソース管理の不可視（見えにくくなる)、マルチテナントでのリソース競合に注意を要する。
・調達するハードウェアが利用したい仮想化機構に対応しているか確認を行う。
・利用したいソフトウェア（プログラム）が仮想環境に対応しているか確認を行う。また、仮想環境ではライセンスの考え方が変わるものもあり確認が必要である。
・利用したいソフトウェア（プログラム）が仮想環境に対応するか確認を行う。
・仮想化に移行する場合、特にデバイスドライバ関連で必ずしもネイティブと同様に動作するとは限らないため、確認を行う。
・物理環境から移行する場合、移行計画（サイジング、動作検証、データ移行、等）が必要である。
・現行ハードウェアリソースの利用率を測定し、仮想化することが適当か否かを判断すべきである。
・仮想化によりサーバを集約するとハードウェア障害の影響範囲が広がるため、可用性について従来以上に配慮する必要がある。

■本手法に関連する対策事例

「教訓Ｔ８」の対策

３．１０　レビュー手法

システム障害を減らすためのソフトウェア開発成果物の品質に主眼を置いたレビューについて述べる（文献 17)。

レビューの効果に関して、欠陥除去コスト（正味棄却コスト）の低減以外に、定性的な効果として開発に必要となる知識の共有や技術移転促進等が挙げられる。

いくつかのレビュー方法が存在するが、実施者は、ソフトウェアライフサイクルの各工程で、プロジェクト特性との整合及びレビュー手法の特徴をとらえたレビュー技法の選択、ならびに自組織にあったやり方をカスタマイズし構築・導入していくことになる。

以下、レビュー手法を紹介する。（８）インスペクション以外は名前のみ。

（１）アドホック・レビュー
（２）ペア・プログラミング
（３）ピア・デスク・チェック
（４）ピア・レビュー
（５）パス・アラウンド
（６）チーム・レビュー
（７）ウォークスルー
（８）インスペクション

インスペクションは、ANSI/IEEE 標準 1028-1998 において体系が定義されているもっとも厳格で公式なレビュー手法である。目的を次のように定めている。

- 成果物（ソフトウェア要素）が基本仕様、指定された品質属性、顧客要求を満たしているか否かの検証。
- 成果物（ソフトウェア要素）が関連する基準、規制、規則、計画、手順に適合しているか否かの検証。
- 発見された欠陥とインスペクションに使用した時間に関する評価指標の供給。
- 表現方法の良し悪しや文体に関する問題の検査は行わない。

これらの８つのレビュー手法に期待される主な効果を一覧にまとめたものを紹介する（表３．１０－１)。この表を参考にして、自組織における最適なレビュー手法を選択することができる。また、システムの重要度、レビューポイントの重要度に応じて、異なるレビュー方法を用いることにより、効果的なレビューを行うことができる。

表３．１０－１　レビュー手法に期待される主な効果[8]

種別＼効果	欠陥製品の発見	仕様との適合性のチェック	標準との適合性チェック	製品の完全性と正しさの検証	理解性と保守性の評価	クリティカルまたは高リスクのコンポーネントの品質の実証	プロセス改善のためのデータ収集	文書品質の測定	製品に関する他のチームメンバの教育	対処法についてのコンセンサスの決定	変更または欠陥修正の正しさの確認	別の対処法の検討	プログラム実行のシミュレーション	レビューコストの最小化
アドホック・レビュー	●													●
ペア・プログラミング	●				●				●	●		●		
ピア・デスク・チェック	●	●	●									●		●
ピア・レビュー	●	●	●	●					●	●	●			
パス・アラウンド	●	●	●		●				●					
チーム・レビュー	●	●	●		●		●		●	●	●			
ウォークスルー	●			●					●	●	●	●	●	
インスペクション	●	●	●	●	●	●	●	●						

■本手法に関連する対策事例

「教訓Ｇ６」「教訓Ｔ５」「教訓Ｔ１３」の対策

[8] IPA「3.2.2 レビュー手法の概要」『高信頼化ソフトウェアのための開発手法ガイドブック～予防と検証の事例を中心に～』2011 年

３．１１　サイレント障害対策

ネットワークシステムにおいて、通常の運用監視の仕組みからは検出されないまま、性能劣化等の現象が発生することを「サイレント障害」と呼ぶ。

運用中にサイレント障害の発生を速やかに検知し、分析・対処につなげるための基本的な取組みは、サービス監視機能を用いる、或いは実際にサービスにアクセスしてみる等の方法により、性能劣化が見られないかをチェックすることである。

基本的な取組みを実践した上で、更なる早期発見、対処を望むサービス提供者においては、監視機能に加えて障害検知、分析のための技術・製品が用いられるようになっている。

参考に、最近登場している、サイレント障害の検知や分析を行うための技術例をいくつか紹介する。

（１）サイレント障害切分けシステム[9]

IP ルータ網監視システムのサブシステムとして開発された「サイレント障害検出機能」と「サイレント障害発生区間特定機能」により、サイレント障害の検出、及び障害個所の特定を容易にした。

（２）インバリアント分析技術の応用[10]

性能情報間の相関関係のうち、平常時に変化しない関係（インバリアント）を自動的に学習してモデル化し、そのモデルと一致しない「いつもと違う」挙動をサイレント障害として検知するツールである。

（３）ビッグデータ分析技術の応用[11]

ネットワーク装置のログ（Syslog）や SNS 上の「つぶやき」情報といったビッグデータをリアルタイムに分析し、障害の早期発見、ひいては予兆検知に利用するツールである。

■本手法に関連する対策事例

「教訓Ｔ１１」「教訓Ｔ２２」の対策

[9] 「大規模な IP ネットワークにおける高精度な障害切り分けシステムの開発」（NTT DOCOMO テクニカル・ジャーナル Vol18 No.1

[10] 「WebSAM の分析技術と応用例～インバリアント分析の特長と適用領域～」（NEC 技報 Vol65 No.2/2012）

[11] 「Syslog と SNS 分析によるネットワーク故障検知・原因分析技術」（NTT 技術ジャーナル 2013.07）

３．１２　パッチ管理

IPA セキュリティセンターは、 政府や企業の経営者、セキュリティ担当者等が、自組織の情報セキュリティ対策を向上させることに役立つ資料として、世界的に評価の高い海外の情報セキュリティ関連文書等の翻訳・調査研究を NRI セキュアテクノロジーズ（株）と共同で行い、その成果を一般に公開している。

その中で、パッチ管理についての米国国立標準技術研究所（NIST）による推奨「パッチ及び脆弱性管理プログラムの策定」を紹介する（文献 18)。

ソフトウェアのバグによるシステム障害を未然に防ぐには、タイムリーなパッチ適用が重要である。特にセキュリティ対策では、より素早い適用が求められている。そのため、米国国立標準技術研究所（NIST）では、各組織において、以下の対策を取るように要求している。

（１）各組織は、組織内部でパッチを特定し、配布を円滑に行うために、パッチ及び脆弱性グループ（PVG）を設置すること。

（２）各組織は、自動化されたパッチ管理ツールを使うことで、システムへのパッチの配布を迅速に行うこと。

（３）各組織は、段階的な方法でエンタープライズ向けパッチ管理ツールを導入すること。

（４）各組織は、エンタープライズ向けパッチ管理ツールの導入に関するリスクを評価し、軽減すること。

（５）各組織は、IT リソースに対する標準化された構成の使用を検討すること。

（６）各組織は、パッチ及び脆弱性管理プログラムの効果を定期的に測定し、必要に応じて是正措置を適用すること。

■本手法に関連する対策事例

「教訓Ｔ６」「教訓Ｔ１６」の対策

３．１３　高回復力システム基盤導入

万が一、情報システムが停止した場合でも、情報システムが一定の強度（停止させないための対策が施されていること）を保つための対策がシステム基盤に導入されていれば、基盤上で実行される業務システムやそれにより提供されるサービスも目標時間内に復旧することが可能になる。

IPA では、事故の発生を前提とした考え方に基づき、IT サービス提供の中断から速やかに復旧するために必要な対策が組み込まれた情報システムのことを「高回復力システム」、高回復力システムにより構成されたシステム基盤を「高回復力システム基盤」と呼んでいる。そして、事業継続戦略の策定やそれを実現する情報システム対策が十分にできていない企業や地方公共団体等が「大規模災害」及び「大規模システム障害」に備えた対応を容易に実施できるための「高回復力システム基盤導入ガイド」（以下、「導入ガイド」とする。）を策定した（文献 19)。導入ガイドでは、これまで費用や専門的スキルをもった要員不足等の理由で、高回復力システム基盤の導入が難しかった組織において、クラウドサービスがその解決策の一つになりうると考え、クラウドサービスを活用した高回復力システム基盤の導入事例等も提供している。

事業継続のための情報システム対策の取り組みにおいては、災害や障害が発生した場合の被害想定や事業への影響度分析を行い、事業継続戦略を決定して、それを実現するためにどのようなシステム基盤が必要かを検討するという手順を踏む。しかし、事業部門で事業継続戦略を決めてくれない、そもそも被害想定や影響度分析、事業継続戦略の決定の方法がわからない、といったケースも少なくない。

導入ガイドでは、高回復力システム基盤のモデルシステムを用意し、経営層や事業部門がモデルシステムの選定を通じて、組織の重要な業務に対してどのようなシステム基盤が必要かを判断することができるようにした。

モデルシステムは、高回復力システム基盤のカタログのようなものである。例えば、自動車の詳しい仕組みは知らなくても、カタログで自動車の性能や装備と、価格を比較して欲しい自動車を選択できるように、IT に詳しくない方でもモデルシステムを参照して自らの組織の要望に適合するシステム基盤を選択することができる（表３．１３－１）。

■本手法に関連する対策事例

「教訓Ｇ５」「教訓Ｔ９」「教訓Ｔ１０」の対策

表３．１３－１　モデルシステムの特性と主要機能[12]

			モデルシステム			
			1	2	3	4
モデルシステムの特徴	①システム基盤の強度		低	中	高	高
	②復旧時間	障害時	１～３日	２時間以内	２時間以内	２時間以内
		災害時	１～６ヶ月	１～６ヶ月	１～７日	２時間以内
	③投資規模		低	中	高	高
モデルシステムの主要要件	①バックアップ方式、取得間隔		非同期 月次	非同期 週次	非同期 数回／日	同期 数回／時
	②機器などの冗長化		なし	あり	あり	あり
	③バックアップサイト		なし	なし	あり	あり （ホット スタンバイ）

[12] IPA『高回復力システム基盤導入ガイド（概要編）』2012 年表 2.3-1　モデルシステムの特徴と主要要件（P.14）

３．１４　ＲＤＢシステム管理

RDB におけるシステム障害対策は、RDB（関係データベース）の運用管理に関する機能を整理し、その機能毎に対策を講じることになる。

IPA/オープンソフトウェア･センタ（OSS）は、人材育成ワーキンググループ（WG）を結成し、OSS の人材育成に取り組んできた中で、2008 年 10 月に、「OSS モデルカリキュラム v1」を公開し、以降、導入実証事業により OSS モデルカリキュラムの普及を図ってきた。そのカリキュラムのひとつである「RDB システム管理に関する知識」が、RDB におけるシステム障害の対策に役立つと考えられる（文献 20)。

データベースを正常に動作させるためには、バックアッププランの策定、リカバリ方法の決定、データベース動作監視といった作業手順を把握することが重要で、これによりデータベース障害に速やかに対応することができる。

特に、データベースが正常に稼働しているかどうかの監視については、以下の点を明確にし、その監視手順を作成し、運用することになる。[13]

- 監視項目
  - * CPU 使用率、メモリ使用率、ネットワーク帯域使用率
  - * ディスク使用率、ディスク I/O 効率、データファイル使用率
  - * ログ出力内容
  - * 表データの断片化
  - * 表データの参照頻度、更新頻度
- 索引の更新頻度
- 監視項目の閾値
- 監視時間帯、監視間隔
- 監視警告通知設定

■本手法に関連する対策事例

「教訓Ｔ１９」の対策

[13] IPA『7-2-基.RDB システム管理に関する知識』2008 年　P.7-2-基-20

３．１５　ヒューマンファクターズ

システム障害の 3 大原因としてハードウェア障害、ヒューマンエラー、ソフトウェア障害があり、その内ヒューマンエラーが約 30%を占めるといった報告がある。[14]

人間はエラーをする（ヒューマンエラー）事が前提にあり、人間自身が安全を意識し、安全を確保するための技量を身につけることは勿論であるが、人間を取り巻くシステム（機械・設備も含む）を改良して安全性を補うことがより重要である。

ヒューマンファクターズとは、人間や組織・機械・設備等で構成されるシステムが、安全かつ経済的に動作・運用できるために考慮しなければならない人間側の要因のことである。[15]

システム開発、運用保守についても、作業手順や作業環境等で、障害を発生させた原因として、それが人間のエラーを引き起こしやすい状況であるかどうかを調査し、改善していく取り組みを積極的に行い、有効な対策を積み重ねていくことが必要である。その対象は「個人」だけでなく、「チーム」や「組織」等の集団としての対策として考える。

2015 年 3 月時点において、IPA でまとめたヒューマンファクターズに関する文献はないが、2014 年のグローバルシンポジウムで、英国 RSSB（鉄道安全標準化機構）のシニアヒューマンファクターズスペシャリスト、ヒュー・ギブソン氏を招いて「ヒューマンエラーとシステム安全」と題して講演を行っている。[16]

また、一般に多数の研究書、実用書が出回っているので、それらを参考にしていただきたい。本教訓集では、「４．３　ヒューマンエラーの問題と対策（詳細説明）」において、報道されたシステム障害のうち、ヒューマンエラーが原因で発生したものについて、詳細な分析を行っている。また、「（別冊）障害分析手法」では、ヒューマンエラーが関係した事象分析手法「２．２　ImSAFER」をもとに、システム障害事例での適用方法を紹介しているので、そちらも参考にしていただきたい。

■本手法に関連する対策事例

「教訓Ｇ４」「教訓Ｇ６」「教訓Ｔ２１」の対策

---

[14] 日常でも、いざという時にも役立つバックアップとは？，ITmedia，2013.3.28
http://www.itmedia.co.jp/enterprise/articles/1303/28/news003.html

[15] “事故・災害の ヒューマンファクターズ” 日本損害保険協会
https://www.sonpo.or.jp/archive/publish/bousai/jiho/pdf/no_223/yj22342.pdf

[16] Huw Gibson 『RSSB and the role of human factors in supporting railway system safety』
https://www.ipa.go.jp/files/000042513.pdf

３．１６　ディペンダビリティ

ディペンダビリティ（dependability）とは、例えば、システムの一部が壊れても残りの部分で補いながら、または機能を縮小させながら、稼働状態を保つといった自立的、自己修復的な動作を示す概念である。[17]

IT サービスがいつでも安心して利用できるためには、それらが利用者や利用者の財産に被害を与えずに提供されるための「安全性」や、提供された IT サービスが定められた提供時間内に常に利用できるための「信頼性」、また、その IT サービスが外部からの侵入・改ざんや情報漏えいを起こさないための「セキュリティ」などの要素が確保されている必要がある。

このような、利用者が安心してシステムを利用し続けるために必要な要素を総合してディペンダビリティ（dependability）と呼び、このディペンダビリティの確保が IT サービスを提供するシステムの重要な課題となっている。

ここで言うシステムの他に、ソフトウェアが組み込まれた製品についても、同様に指摘することができる。

■本手法に関連する対策事例

「教訓Ｔ２４」の対策

[17] JIS Z 8115：2000 では、ディペンダビリティは、「アベイラビリティ性能及びこれに影響を与える要因、すなわち信頼性性能、保全性性能及び保全支援能力を記述するために用いられる包括的な用語。（備考１．ディペンダビリティは非定量的用語として一般的記述に限り用いられる）」

３．１７　原因不明時の対策

システム障害発生時にシステムの復旧を優先させるあまり、原因調査のデータ（メモリ情報、トレース、ログなど）を保存する作業を行わず、システムの再起動を行ってしまい、復旧後に原因調査を行おうとした場合に、原因判明にまで到達しない事態が起きることがある。

そのような場合は、次にシステム障害が起きた時に、原因判明に必要な情報をいかに漏れなく採取するかを、対策の中に組み込むことが重要である。

そのためにも、「なぜ障害原因を特定できなかった」のかについて、「なぜなぜ分析」などを行い、次にシステム障害が起きた時の対策を検討する。

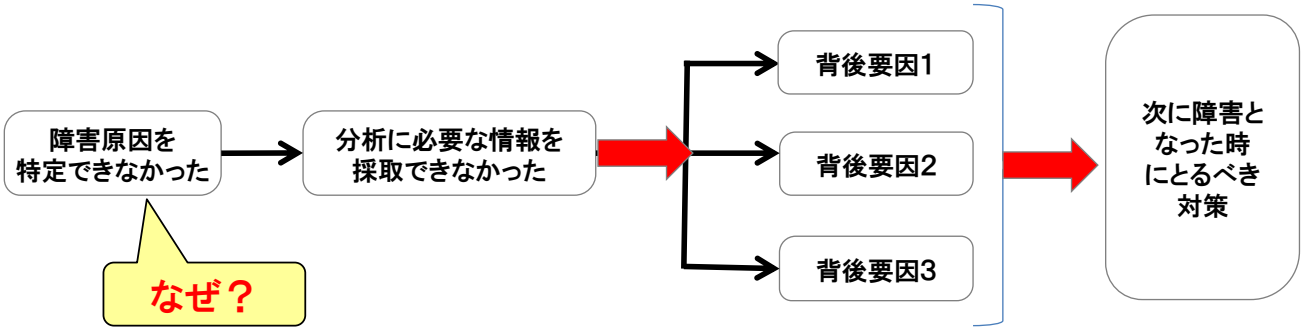


図３．１７－１　なぜなぜ分析例

■本手法に関連する対策事例

「教訓Ｔ２５」の対策

３．１８　周期処理の対策

制御システム（組み込みシステム）の中で、しばしば活用されるプログラムロジックとして周期処理がある。周期処理は、ひとつの完結する処理群が周期時間内に完了することが条件であり、新たな機能や、制御すべき機器が増えた場合でも、その周期内で処理が完了することは必須である。

IPA では、組込みソフトウェアの開発現場で、設計を行う上で求められる品質を確保したシステム・ソフトウェアを実現するために現場で行われている設計上の工夫、注意点を事例として収集し、それぞれの事例からノウハウを抽出し、設計方法と整理文書化したガイド（事例集）を書籍として公開している（文献 21)。この書籍から、周期処理の設計に関係する箇所を紹介する。

A-12　イベントを周期的に処理する場合は周期を意識した安易な処理分割をしない[18]

作法概要：・周期的なイベント処理では、受信したイベントに関する処理は、ポーリング周期内に完了させる。
・その処理が周期内に完了しない場合でも、安易に処理分割しない。

メリット：　保守性（解析性、安定性、試験性）を維持することができる。

留意点　：　ポーリング周期内にどうしても処理が完了しない場合には、設計の見直しを行う。

A-17　時間制約を定義しこれを守ることにより確実な動作を保証する[19]

作法概要：・各ソフトウェア機能項目に時間制約を定義する。
・同じ時間制約を持つ機能項目を同じタスクに割り付ける。

メリット：　タスク設計において処理性能のバラツキを抑制させ、保守性や信頼性、効率性を確保することができる。

留意点　：・時間制約は、物理的な特性や制御対象物の特性から定義し、タスクを周期的に起動することで時間制約を満足させるようにする。
・周期を定義できない場合は、イベント発生から起動開始までの時間制約を定義する。

C-18　機能変更を行う前にパフォーマンスに与える影響を十分検討する[20]

作法概要：・部分的なソフトウェアの変更でも他の部分に影響し得ることを認識する。
・機能追加または変更に伴うパフォーマンスへの影響について、事前に十分検討する。

メリット：　機能追加または変更に伴う影響を緩和することができる。

留意点　：　変更が他の部分へ影響を与える場合、影響があったとしても変更を実施するのか、または実施しないのかは十分に検討する。

■本手法に関連する対策事例

「教訓Ｔ３２」の対策

[18] IPA「組込みソフトウェア向け設計ガイド　ESDR[事例編]」2012 年　(P.44)
[19] IPA「組込みソフトウェア向け設計ガイド　ESDR[事例編]」2012 年　(P.59)
[20] IPA「組込みソフトウェア向け設計ガイド　ESDR[事例編]」2012 年　(P.171)

３．１９　レースコンディションの対策[21]

レースコンディションとは、並列動作する複数の存在（プロセスやスレッド）が同一のリソースへほぼ同時にアクセスしたとき、予定外の処理結果が生じる問題である。

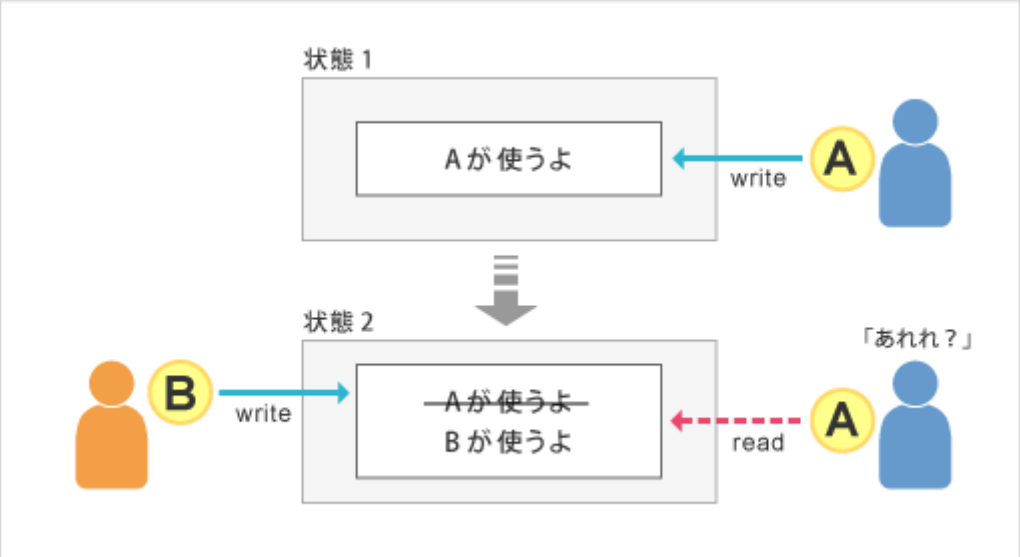


図３．１７－１　リソースへの同時アクセス

データベース更新トランザクションの競合も、そのひとつの例である。あるトランザクションが一つのレコードを読み取って処理し新しい値を書き戻そうとしているとき、その書き込みが終わる前に別のトランザクションが更新の目的で同じレコードからの読み出しを始めると、結果としてこのレコードには不適切な値が書き込まれることになる。システム開発の現場では、このような不具合が起こらないようにする機能を、排他制御と呼ぶ。

レースコンディションによって、情報漏えいを防ぐためには、「排他制御」機能の設計時、検証時の対応を十分検討する必要がある。

■本手法に関連する対策事例

「教訓Ｔ３３」の対策

21 セキュアプログラミング講座「第 4 章 不測の事態対策レースコンディションの一般的対策」

４．おわりに

障害事例の分析によって得られた教訓に対して、有効な対策手法をリストアップし、概要を述べた。

今後とも、教訓の増加に伴って、それに対応する対策手法も追加していくとともに、分類整理をより精緻化していく予定である。

また、ソフトウェア・エンジニアリングからの新しい技術についても、対策手法として活用できるものは、追加していく予定である。

情報処理システム高信頼化 教訓集

IT サービス編　別冊Ⅰ：障害対策手法

2019 年 2 月 28 日 PDF 版発行
2020 年 3 月 16 日 改訂

監修者 独立行政法人情報処理推進機構（IPA）
社会基盤センター
発行人 片岡　晃
発行所 独立行政法人情報処理推進機構（IPA）
〒113-6591
東京都文京区本駒込二丁目 28 番 8 号
文京グリーンコート センターオフィス
https://www.ipa.go.jp/ikc/

独立行政法人 情報処理推進機構
社会基盤センター